ISSN 0860-1941

CENA 220 zł

mikroklan

vademecum techniki mikrokomputerowej

Scanned by Biken

Zeszyt 3

Pamięć dla Z80

Co to jest CAD/CAM?

Klawiatura hallotronowa

Czy AMSTRAD 6128

jest komputerem osobistym?

CAD/CAM

WYDAWNICTWO NOT SIGMA

Kształtujmy wspólnie profil Mikroklanu!

Oczekujemy
ciekawych rozwiązań
niestandardowych aplikacji
wymiany doświadczeń

Przesyłajcie propozycje i opinie na nasz nowy adres:
01-517 Warszawa, ul. Mickiewicza 18 m. 17
tel. 39 14 34

mikroklan ZESZYT 3

Wprowadzenie
w zagadnienia
techniki
mikrokomputerowej

Praca zbiorowa

Opracował zespół redakcji
miesięcznika INFORMATYKA

Teresa Jabłońska, Władysław Klepacz, Ireneusz Myzik, Jerzy Orkiszewski, Maria Pawlak, Andrzej J. Piotrowski, Adam Pluta, Zbigniew Pojmański, Mateusz Stryjecki (oprac. graficzne)

WYDAWNICTWO CZASOPISM I KSIĄŻEK TECHNICZNYCH

SIGMA

PRZEDSIĘBIORSTWO NACZELNEJ ORGANIZACJI TECHNICZNEJ

00-950 Warszawa skrytka 1004
ul. Biała 4

ISSN 0860-1941
Wydawnictwo Czasopism
i Książek Technicznych
NOT-SIGMA
Warszawa 1986

Redakcja: 01-517 Warszawa,
ul. Mickiewicza 18 m. 17, tel. 39 14 34
Skład: technika fotoskładu Eurocat 150
Wydawnictwo NOT-Sigma
Druk: Bohmann Druck und Verlag
GmbH&Co. KG, Wiedeń, Austria
Nakład 100 000 egz. P-47
Drukowano na zlecenie ARS POLONA
Cena 220 zł

SPIS TREŚCI

SPRZĘT

- Klawiatura hallotronowa
  Jerzy Orkiszewski, Ignacy Nowosielski 15
- Pamięć dynamiczna dla procesora Z80
  Piotr Smólski 20
- Pamięć zewnętrzna na minidyskach elastycznych
  Adam Pluta 22

OPROGRAMOWANIE

- Czym jest UNIX (3). Polecenia systemowe (cd.)
  Janusz Zalewski 8

ZASTOSOWANIA

- Konstruowanie przestało być trudne
  micro 5
- Co to jest CAD/CAM
  micro 10
- Mikrokomputer zamiast deski kreślarskiej
  micro 30

INFORMATYCZNE ABC

- BASIC dla początkujących
  micro 2

EDUKACJA

- Komputer czy cyrkiel i linijka
  Barbara Chrzan-Feluch 28

OPIS

- Komputer osobisty DIALOG DTC-8
  Krzysztof Kamiński 12
- Amstrad CPC 6128
  micro 21
- System dyskowy MUEL-85
  Piotr Parlewicz, Andrzej Więckowski, Grzegorz Zawadzki 26

BASIC dla początkujących

W dwóch pierwszych odcinkach przedstawiliśmy podstawowe zagadnienia związane z komputerem: tryby pracy, zmienne oraz redagowanie (edycja) programu. Omówiliśmy ważne instrukcje: INPUT, PRINT. Krótkie powtórzenie tych zagadnień znajdziesz w ramce. W obecnym odcinku zajmiemy się niezwykle istotnym elementem sztuki programowania – pętlą programową. Opiszemy też nie mniej istotną instrukcję warunkową. Umożliwia ona rozgałęzienia programu zależne od dotychczasowego przebiegu.

Kurs jest przeznaczony dla użytkowników komputera Commodore 64, jednak bez większych trudności może być wykorzystany przez użytkowników innych komputerów wyposażonych w język BASIC.

W poprzednim odcinku przedstawiliśmy program obliczający pierwiastek, kwadrat i sześcian wprowadzonej liczby. Jak zmienić ten program, aby otrzymać wyniki dla wielu kolejnych liczb naturalnych? Wprowadź i wykonaj program przedstawiony na wydruku 1. Gdy będziesz chciał go przerwać, wciśnij klawisz *STOP*.

Wydruk 1

```
10  INPUT "LICZBA WPROWADZANA";L
20  P = SQR(L)
30  K = L*L
40  S = L*L*L
45  PRINT "ARGUMENT",L
50  PRINT "PIERWIASTEK",P
60  PRINT "KWADRAT",,K
70  PRINT "SZESCIAN",S
75  PRINT
80  PRINT L = L + 1
90  GOTO20
```

Modyfikacja programu polegała głównie na dodaniu linii 80 i 90. W linii 80 następuje zwiększenie o 1 wartości argumentu, dla którego przeprowadzane są obliczenia. Linia 90 zawiera nową instrukcję: instrukcję skoku **GO TO** (z numerem 20). **GO TO** (czytaj goł tu), to w języku angielskim *idź do*. Po wykonaniu instrukcji **GO TO** następuje skok do linii o podanym numerze (w tym wypadku do linii 20). A więc zamiast kolejnej linii (której tu nie ma, ale która może występować w innych programach) wykonywana jest zupełnie inna linia. Następuje skok. Wszystkie linie programu, począwszy od 20, zostaną wykonane ponownie. Ponieważ zmieniliśmy wartość argumentu **L**, otrzymamy nowe wartości pierwiastka, kwadratu i sześcianu dla kolejnej liczby naturalnej. Obliczenia te będą powtarzane od nowa dla kolejnych liczb naturalnych – aż do chwili, gdy wyłączymy komputer lub przerwiemy program, wciskając klawisz *STOP*.

INSTRUKCJE WARUNKOWE

Często obliczenia, podobne do występujących w programie wydruku 1, chcemy wykonać dla argumentów zawartych w ustalonym przedziale liczbowym. Trzeba więc wprowadzić wartość końcową – górny kraniec przedziału liczbowego, z którego wyznaczamy argumenty obliczeń. Wprowadź:

15 INPUT „WARTOSC KONCOWA ARGUMENTU", WK

W programie trzeba sprawdzić, czy argument osiągnął już wartość końcową. W nowej linii (przypiszmy jej numer 76) należy wykonać następujące działanie: *JESLI argument osiągnął wartość końcową, TO przerwij program.*

Jak przerwać program? Służy do tego instrukcja **END**. Może ona występować w wielu miejscach programu i oznacza logiczny koniec programu. Instrukcja **END** nie musi być ostatnią instrukcją w programie.

Jak sprawdzić warunek? Nasze zdanie: *JESLI... TO... trzeba zapisać w angielskiej formie IF... THEN...*. Po **IF** (jeśli) należy podać warunek, jaki musi być spełniony, a po **THEN** (to) akcję (instrukcję) jaką należy podjąć w wypadku spełnienia tego warunku. W wypadku niespełnienia warunku nie zostaje podjęta żadna akcja – po prostu następuje przejście do wykonania kolejnej instrukcji. Warunek do spełnienia (argument równy wartości końcowej) zapisujemy w postaci **L = WK**, zaś akcja, jaką należy podjąć w wypadku spełnienia warunku, to zakończenie programu, czyli instrukcja **END**. Instrukcja sprawdzająca warunek i ewentualnie podejmująca akcję, nosi nazwę instrukcji warunkowej. Wprowadź naszą instrukcję warunkową:

76 IF L = WK THEN END

Wykonaj program. Dla wszystkich danych w wybranym zakresie (większych lub równych wartości początkowej i mniejszych lub równych wartości końcowej) zostały przeprowadzone obliczenia. Po wykonaniu obliczeń komputer wykonał instrukcję **END** i przeszedł do trybu bezpośredniego – na ekranie pojawił się znany nam już komunikat **READY**.

SCHEMATY BLOKOWE PRZEBIEGU PROGRAMU

W ramach naszego kursu, oprócz opisu samych instrukcji języka, chcemy przedstawić pewne dodatkowe, użyteczne informacje. W tym momencie warto nadmienić o schematach blokowych przebiegu programu.

Schematy blokowe są pomocniczym środkiem przy projektowaniu i dokumentowaniu programu. Dla małych programów nie zawsze są one stosowane, co zresztą nie jest dobrą praktyką. Przy dużych programach i pakietach programowych, pisanych z reguły przez wielu ludzi, schematy blokowe stają się koniecznością. Przed napisaniem instrukcji trzeba bowiem ustalić sens i cel programu oraz jego przebieg.

Można to zrobić, na przykład w formie schematów blokowych. Gdy program będzie gotowy, wówczas schematy blokowe stanowić będą dokumentację sposobu jego działania. Schematy blokowe czy sieci działań spotyka się w wielu innych dziedzinach, nie są one więc jakąś szczególną cechą dyscypliny programowania. Schematy blokowe obrazują cały przebieg programu. Wydruk (listing) programu jest wprawdzie uporządkowany według numerów poszczególnych linii, niemniej program nie musi wcale przebiegać przez kolejne linie. Widzieliśmy to już na przykładzie instrukcji **GO TO**.

W schematach blokowych stosuje się pewne powszechnie przyjęte symbole: początek i koniec programu jest oznaczany za pomocą kółek, normalny przebieg programu za pomocą prostokątów, zaś rozgałęzienie za pomocą rombów (albo trapezów). Nasz krótki program możemy więc przedstawić w postaci schematu blokowego, pokazanego na rysunku. Na schemacie tym wyraźniej widać sens instrukcji warunkowej **IF... THEN...**

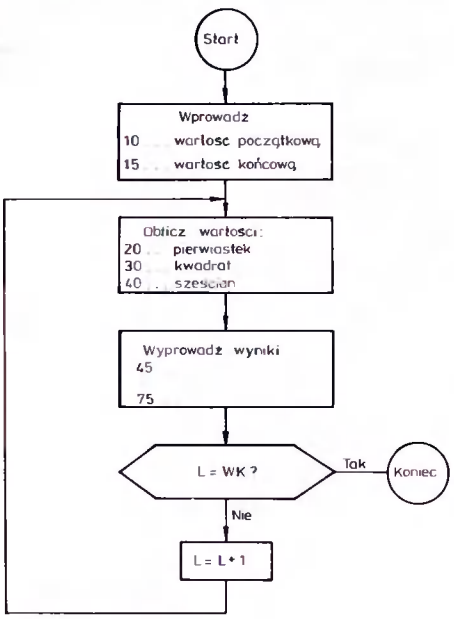
Zapamiętaj:

Za pomocą instrukcji warunkowych można sprawdzać stan programu i w zależności od niego – wykonywać (lub nie wykonywać) określone instrukcje. Instrukcja warunkowa ma postać: IF (warunek) THEN (instrukcja wykonywana w przypadku spełnienia warunku).

Wydruk 2

```
10  INPUT "LICZBA WPROWADZANA";L
15  INPUT "WARTOSC KONCOWA ARGUMENTU",WK
20  P = SQR(L)
30  K = L*L
40  S = L*L*L
45  PRINT "ARGUMENT",L
50  PRINT "PIERWIASTEK",P
60  PRINT "KWADRAT",,K
70  PRINT "SZESCIAN",S
75  PRINT
76  IF L = WK THEN END
80  L = L + 1
90  GOTO20
```

Na wydruku 2 pokazany został skorygowany program.



Schemat blokowy programu

PĘTLE PROGRAMOWE

Warunkowe powtarzanie wykonywania instrukcji można też zrealizować w nieco inny sposób: za pomocą pętli programowej. W naszym programie linie od 20 do 90 były wykonywane dopóty, dopóki w linii 76 nie nastąpiło wykrycie osiągnięcia przez argument **L** wartości końcowej **WK**. Nasze zadanie można by sformułować następująco: **„od wartości początkowej aż do wartości końcowej obliczaj...”**. Do realizacji tak postawionego zadania, czyli do wykonywania pętli programowej, służą występujące zawsze parami instrukcje **FOR** i **NEXT**, które tworzą konstrukcje **FOR... NEXT**.

Wprowadź:
18 FOR I=L TO WK
80 NEXT

Następnie w wierszach od 20 do 45 zastąp zmienną **L** przez zmienną **I**. W tym celu wprowadź najpierw polecenie **LIST**, co spowoduje, że na ekranie pojawi się treść programu. Za pomocą klawiszy ze strzałkami (i ewentualnie klawisza SHIFT) przesuń kursor na literę **L** i wciśnij klawisz **I** . **L** zostanie zastąpione przez **I**. Po dokonaniu wszystkich potrzebnych zmian w danej linii wciśnij klawisz *RETURN* – komputer wykona wtedy polecenie zapisane w tej linii, czyli wprowadzi do pamięci linię w nowej postaci. Postępuj tak ze wszystkimi zmienianymi liniami, po czym przesuń kursor poniżej zapisanej części ekranu. Na koniec zlikwiduj linie 76 i 90 (w celu zlikwidowania danej linii wprowadź jej numer i wciśnij klawisz *RETURN*). Wprowadź polecenie **LIST** – na ekranie pojawi się nowa wersja programu – wydruk 3.

Wydruk 3

```
10   INPUT "LICZBA WPROWADZANA";L
15   INPUT "WARTOSC KONCOWA",WK
18   FOR I = L TO WK
20   P = SQR(I)
30   K = I*I
40   S = I*I*I
45   PRINT "ARGUMENT",I
50   PRINT "PIERWIASTEK",P
60   PRINT "KWADRAT",,K
70   PRINT "SZESCIAN",S
75   PRINT
80   NEXT
```

Skupmy się teraz na pętli programowej. Ma ona trzy parametry:

- zmienną kontrolną pętli
- wartość początkową zmiennej kontrolnej pętli
- wartość końcową zmiennej kontrolnej pętli.

Zapamiętaj:

Wartością początkową oraz końcową zmiennej kontrolnej pętli może być zarówno stała liczbowa, jak i inna zmienna.

W naszym programie na wartość początkową wybraliśmy zmienną **L**, zaś na końcową – zmienną **WK**. Początek pętli programowej mógłby też mieć postać:
18 FOR I = 1 TO 10
Za zmienne kontrolne pętli można przyjmować zmienne I, J, K... .

Zapamiętaj:
Koniec pętli zaznaczany jest za pomocą instrukcji NEXT.

Co właściwie dzieje się w pętli programowej? W linii 18 zmiennej kontrolnej pętli **I** przypisywana jest wartość początkowa **L** i definiuje się wartość końcową **WK**. Linia 80 (*NEXT*) ma za zadanie sprawdzić, czy zmienna kontrolna pętli osiągnęła już wartość końcową (warunek **IF** w poprzedniej wersji programu) oraz automatycznie zwiększyć o 1 zmienną kontrolną pętli (instrukcja **L=L+1** w poprzedniej wersji). Na koniec linia 80 (*NEXT*) w wypadku osiągnięcia przez **I** wartości końcowej powoduje wykonanie skoku do instrukcji 20, albo w wypadku przeciwnym – przejście do instrukcji następnej po *NEXT*. Jeśli nie ma takiej instrukcji, następuje zakończenie programu i przejście do trybu bezpośredniego.

Zapamiętaj:
Pętla programowa po wejściu do niej zawsze będzie wykonana przynajmniej jeden raz.

Dzieje się tak także wtedy, gdy wartość początkowa jest większa od wartości końcowej:
100 FOR I = 2 TO 1
...
200 NEXT

PĘTLE ZAGNIEŻDŻONE

W czasie wykonywania naszego programu trudno było obserwować wyniki, gdyż zbyt szybko przesuwały się one po ekranie. Znamy już pętlę programową i wiemy, że jest ona jedynie skróconą formą zapisu instrukcji nakazujących komputerowi sprawdzenie warunku zakończenia wykonywania pętli, zwiększanie zmiennej kontrolnej pętli i warunkowe wykonanie skoku na początek pętli. Stworzymy teraz dodatkową pętlę programową do spowalniania wyświetlania informacji. Wprowadźmy tzw. pętlę oczekiwania.
Wprowadź:
76 FOR J=1 TO 1000
77 NEXT
Jest to pętla zagnieżdżona w poprzedniej pętli. W pętli pomiędzy liniami 76 i 77 właściwie nic się nie dzieje poza tym, że komputer zużywa czas. I o to właśnie chodzi! Wykonaj program. Okazuje się, że po obliczeniu wartości dla każdego kolejnego argumentu komputer czeka przez chwilę. Spowodowała to pętla oczekiwania. Czas trwania pętli oczekiwania możesz ustalać poprzez zmianę różnicy między wartością końcową i początkową zmiennej kontrolnej pętli oczekiwania.

Zapamiętaj:
Pętle oczekiwania można zrealizować za pomocą „pustych” pętli FOR... NEXT
Zapamiętaj:
Pętle mogą być zagnieżdżone jedna w drugiej. Dla każdej pętli trzeba jednak użyć innej zmiennej kontrolnej pętli. Wartości początkowe i końcowe mogą być takie same.

Na koniec uwaga dotycząca używania zmiennej kontrolnej w programie.
Zapamiętaj:
Po zakończeniu pętli zmienna kontrolna ma wartość równą wartości końcowej zwiększonej o 1.

DANE TESTOWE NA EKRANIE

Wykonaj ostatnią wersję programu i po komunikacie „READY” wprowadź:
? I,J
Po naciśnięciu klawisza *RETURN* na ekranie pojawią się wartości 11 i 1001.
Zapamiętaj:
Wartości zmiennych używanych w trybie wykonywania programu można wyświetlać w trybie bezpośrednim za pomocą rozkazu PRINT lub jego skróconej formy (?).

W ten sposób zetknąłeś się z podstawową zaletą interpretera. Co to jest interpreter? Interpreter interpretuje w czasie wykonywania programu instrukcje napisane przez Ciebie w języku BASIC. To znaczy, komputer tłumaczy Twój program podczas jego wykonywania na swój kod maszynowy (zera i jedynki) i od razu wykonuje ten kod, realizując w ten sposób poszczególne instrukcje.

Przeciwieństwem interpretera jest kompilator. Tłumaczy on instrukcje tworzące napisany przez Ciebie program na kod maszynowy i zapamiętuje go, lecz nie wykonuje. Kod ten (a więc Twój program) może być wykonany później, po zakończeniu procesu tłumaczenia zwanego kompilacją.

Zapamiętaj:

Przy pracy z interpreterem, w każdej chwili można zatrzymać wykonywanie programu, w celu obejrzenia chwilowych wartości zmiennych.

Wykonaj jeszcze raz nasz program, np. dla wartości od 1 do 100. W pewnej chwili wciśnij klawisz STOP – komputer pokaże wtedy, w której linii przebieg programu został przerwany. Wprowadź teraz

? I,J,L,WK,P,K,S

i wciśnij klawisz RETURN. Komputer wyświetli wartość chwilową zmiennych w momencie przerwania programu. Mogą one np. wyglądać następująco:

7	**350**	**1**	**100**
2.64575131		**49**	**343**

Oznacza to, że w pętli zewnętrznej (zmienna kontrolna I) przeprowadzane są obliczenia dla argumentu 7 oraz że zmienna kontrolna J w wewnętrznej pętli oczekiwania (linie 76 i 77) osiągnęła wartość 350.

Tę okoliczność możesz wykorzystać przy większych programach. W czasie przebiegu programu możesz wyświetlić wartość pewnych zmiennych i porównać wartości spodziewane z wartościami rzeczywistymi. Z tego powodu łatwiej jest testować programy interpretowane niż programy, które muszą być najpierw skompilowane.

PĘTLE Z WARTOŚCIĄ KROKU

Aby umożliwić stosowanie pętli nie tylko dla kolejnych liczb naturalnych, instrukcja pętli może występować w formie z podaną wartością kroku.

Zapamiętaj:

Instrukcja pętli może zawierać wartość kroku. Po każdym przebiegu pętli wartość kroku jest dodawana do zmiennej kontrolnej pętli. Instrukcja pętli z wartością kroku ma postać FOR... TO... STEP... Jeśli nie podano słowa STEP i wartości kroku, jest ona automatycznie przyjmowana za równą 1.

Wartość kroku może być także liczbą niecałkowitą.

Wprowadź nową wersję linii 18

18 FOR I = 10 TO 100 STEP 10

oraz zlikwiduj linie 10 i 15. Zmieniony program pokazano na wydruku 4. Wykonaj program. Argument przyjmuje wartości 10, 20 ..., 100.

Wydruk 4

```
 18  FOR I = 10 TO 100 STEP 10
 20  P = SQR(I)
 30  K = I*I
 40  S = I*I*I
 45  PRINT "ARGUMENT",I
 50  PRINT "PIERWIASTEK",P
 60  PRINT "KWADRAT",,K
 70  PRINT "SZESCIAN",S
 75  PRINT
 76  FOR J = 1 TO 1000
 77  NEXT
 80  NEXT
```

Ćwiczenie: Jesteś rentierem żyjącym z odsetek. Zatem wzór na kwoty wydawane przez Ciebie na życie ma postać:

kwota = kapitał * czas * oprocentowanie/100

Oblicz kwotę zużytą przez Ciebie dla wszystkich wartości kapitału pomiędzy 1000 a 20 000 (z krokiem co 1 000), dla oprocentowania od 5% do 10% i dla upływu czasu od 10 do 20 lat. Wykorzystaj trzy zagnieżdżone w sobie pętle FOR ... NEXT... (rozwiązanie w następnym odcinku).

micro

POWTÓRZENIE WIADOMOŚCI Z CZĘŚCI 1 i 2

* Komputer Commodore 64 może znajdować się w jednym z trzech trybów:
  - trybie bezpośrednim
  - trybie edycji
  - trybie wykonywania programu.
* Wprowadzanie należy kończyć wciśnięciem klawisza *RETURN.*
* Klawisz *INST* służy do przygotowania miejsca na wprowadzenie dodatkowego znaku.
* Klawisz *DEL* służy do wymazania znaku.
* Instrukcja **PRINT** (w formie skróconej: ?) służy do wyprowadzania informacji.
* Instrukcja **INPUT** umożliwia wprowadzanie z klawiatury danych używanych później w programie.
* Instrukcja **LIST** umożliwia wyświetlenie programu na ekranie.
* Instrukcja **RUN** umożliwia uruchomienie zapamiętanego programu.
* Po instrukcji **INPUT** przed nazwą zmiennej można wprowadzić ograniczony z obu stron znakami cudzysłowu tekst opisujący zmienną, której wartość ma być wprowadzana. Pomiędzy tekstem i zmienną musi być średnik (;).
* Zmienne są miejscami w pamięci służącymi do przechowywania wartości liczbowych lub ciągów znaków przypisanych w danej chwili tym zmiennym.
* Stałe to wartości bezpośrednio występujące w programie, przy czym stałe znakowe (ang. strings) muszą występować pomiędzy znakami cudzysłowu (").
* Opis zmiennej w instrukcji **INPUT** nie może być zmienną (nawet znakową).
* W jednej i tej samej instrukcji **PRINT** mogą występować razem zmienne i stałe znakowe i liczbowe.
* Dla dobrego udokumentowania programu, nazwy zmiennych powinny odzwierciedlać ich znaczenie.
* Stałe znakowe mogą zawierać na początku, w środku i na końcu również znaki puste (spacje).
* Przecinek w instrukcji **PRINT** oznacza przesunięcie miejsca wyprowadzania na kolejną pozycję tabulacyjną, nawet gdy między dwoma przecinkami nie znajduje się zmienna ani stała.
* Polecenie uruchamiające program (**RUN**) może znajdować się także wewnątrz programu. Program rozpoczyna się wtedy od początku, zaś wszystkie zmienne mogą ulec zmianie.
* Klawisz CTRL powoduje spowolnienie wyprowadzania na ekranie monitora.
* Klawisz stopu umożliwia przerywanie programu. Pojawia się wówczas meldunek: *BREAK IN* (wiersz).
* CONT zapewnia kontynuację działania programu.
* Pojedyncza instrukcja **PRINT** powoduje przesunięcie linii.

DLA UŻYTKOWNIKÓW ZX SPECTRUM

Język BASIC zaimplementowany na mikrokomputerze ZX Spectrum różni się w pewnych szczegółach od języka BASIC dla mikrokomputera Commodore 64:

* w ZX Spectrum nie ma instrukcji **END.** Zamiast niej może być użyta instrukcja **STOP**
* nazwą zmiennej kontrolnej pętli może być dowolna litera **(a...z)**
* po instrukcji **NEXT** należy koniecznie podać nazwę zmiennej kontrolnej pętli (np. **NEXT i**)

SŁOWNICZEK

GO TO (***gou tu***) – idź do...
END (***end***) – koniec
IF (***if***) – jeżeli
THEN (***dzen***) – to
FOR (***for***) – dla
TO (***tu***) – do
NEXT (***nekst***) – następny (obieg pętli)

Konstruowanie przestało być trudne

Od niedawna oferowane są systemy CAD, które mogą współpracować także z mikrokomputerami. Jednym z takich systemów jest **Caddy** – efektywny program programowania dwuwymiarowego, specjalnie przystosowany do potrzeb małych i średnich biur konstrukcyjnych. Pokażemy na czym polegają zalety tego programu i do jakich zastosowań jest on przeznaczony.

Konstruktorzy urządzeń technicznych i kreślarze musieli wiele czasu poświęcać na rysowanie powtarzających się elementów, wprowadzanie zmian i wymiarowanie rysunków. Te rutynowe czynności można za pomocą programu Caddy w znacznym stopniu zautomatyzować. Dowolny standardowy element może być wykreślony w ciągu paru sekund. Staranne ręczne wykonanie tej czynności zajęłoby wiele minut, a nawet godzin.

SPRZĘT

Caddy jest programem uniwersalnym. Istnieją wersje przeznaczone dla mikrokomputerów Sirius oraz IBM PC/XT i modeli z nim kompatybilnych. Dopasowanie do innych systemów jest też łatwe, pod warunkiem, że mają one następujące cechy:

- pamięć operacyjna co najmniej 384 KB,
- współprocesor arytmetyczny 8087,
- pamięć na dysku sztywnym,
- monitor ekranowy o dobrej rozdzielczości ze sterownikiem,
- interfejsy dla plotera oraz digitajzera (koordynatometru).

Na specjalne podkreślenie zasługuje wykorzystanie standardowego mikrokomputera, na którym można wykonywać także inne prace, jak: przetwarzanie tekstów, prowadzenie księgowości czy sporządzanie list płacy.

DZIEDZINY ZASTOSOWAŃ

Opracowując programy **Caddy** wzięto pod uwagę fakt, że wymagania i potrzeby poszczególnych dziedzin są różne. Najważniejszą cechą tego programu jest zdolność do dalszego przetwarzania informacji wywodzących się z rysunku.

Tak więc **Caddy** nadaje się do zastosowania w elektrotechnice i elektronice, do kreślenia rysunków konstrukcyjnych dowolnego rodzaju. Z uzyskanego rysunku można następnie wygenerować zestawienia elementów i plany montażowe układów. Oznacza to znaczne ułatwienie dla konstruktora.

Za pomocą tego programu, znacznie szybciej niż na desce kreślarskiej, może pracować również architekt. Na podstawie rysunku rzutu podstawowego tworzone są rysunki poszczególnych elewacji budynku, a wymiarowanie rysunków odbywa się szybko i bezbłędnie. Dane z pakietów graficznych mogą być również wykorzystane do wszelkiego rodzaju obliczeń i opisów projektu.

Wysoka dokładność programu i liczne funkcje wspomagające znacznie ułatwiają konstruowanie maszyn. Dzięki rozbudowanym możliwościom powiększania powstają warunki do bardzo precyzyjnego opracowywania detali. Istnieje – spotykana na ogół jedynie w znacznie droższych systemach – możliwość tworzenia kilku wariantów projektu. Oczywiście istnieje również możliwość zastosowania niezależnego od specyfiki wymienionych dziedzin, a więc wszędzie tam, gdzie potrzebne są schematy blokowe, organigramy i wszelkiego rodzaju rysunki.

SPOSÓB PRACY

Program jest tak skonstruowany, że nawet użytkownik bez doświadczenia w dziedzinie techniki komputerowej może bez problemów rozpocząć jego stosowanie. Zasadą jest stosowanie hierarchicznej techniki menu, polegającej na tym, że w każdym kroku pracy użytkownik ma do dyspozycji jedynie te polecenia, które są potrzebne w danym etapie konstruowania. Każdej pozycji menu głównego odpowiada menu podrzędne, które pojawia się na ekranie po naprowadzeniu na nią kursora.

Dane są wprowadzane za pomocą digitajzera i myszki. Urządzenia te służą jednocześnie do wybierania odpowiednich propozycji z menu. Wybranie następuje poprzez każdorazowe wciśnięcie przycisku, znajdującego się na myszce. Jedynie wprowadzanie tekstów, np. do opisu rysunku, wywoływanie symboli oraz rysunków odbywa się za pośrednictwem klawiatury. Użytkownik ma więc zapewniony komfort pracy i nie musi

wciąż przenosić się od klawiatury do digitajzera i z powrotem.

FUNKCJE MENU

Aby ułatwić konstruktorowi korzystanie z programu w procesie konstruowania, uwzględniono specyfikę różnych dziedzin: Przykładowo – dla zapoznania się ze sposobem działania programu – prześledźmy przebieg projektowania rzutu poziomego budynku.

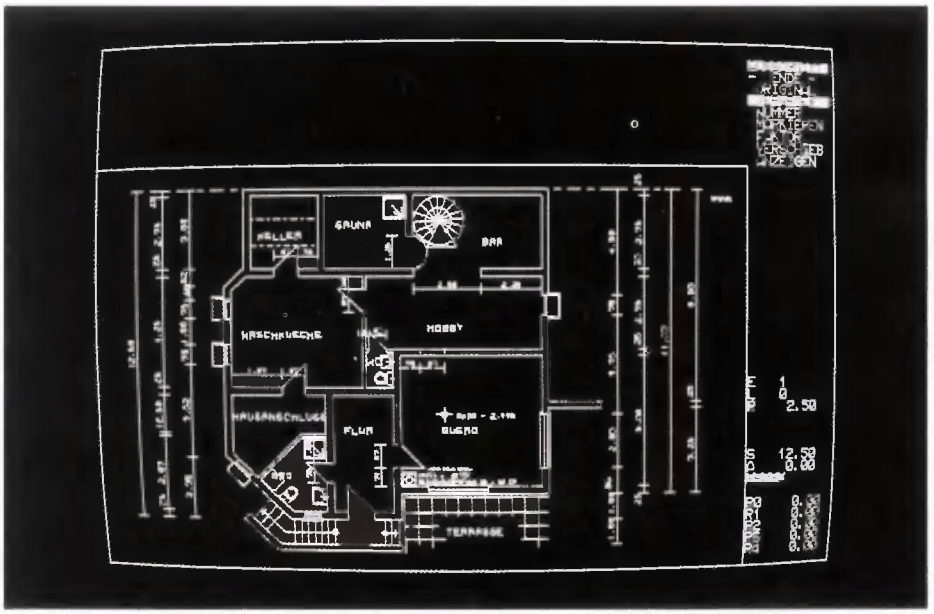
Przy opracowaniu nowej konstrukcji, najpierw określa się granice rysunku, podając np. znormalizowany jego format. W tym celu w menu głównym wybiera się polecenie *„stałe"* (niem. Konstanten). Wymiary rysunku są określone w menu głównym za pomocą szeregu cyfr. Dla optymalnej pracy celowe jest wybranie w każdym kroku określonej siatki (rastru), co ułatwia kreślenie linii poziomych i pionowych, które z powodu dużej rozdzielczości digitajzera trudno byłoby ustawiać na właściwej pozycji w inny sposób. Użytkownik ma do wyboru 16 różnych siatek. Po wybraniu siatki na ekranie pojawiają się odpowiednie punkty rastrowe.

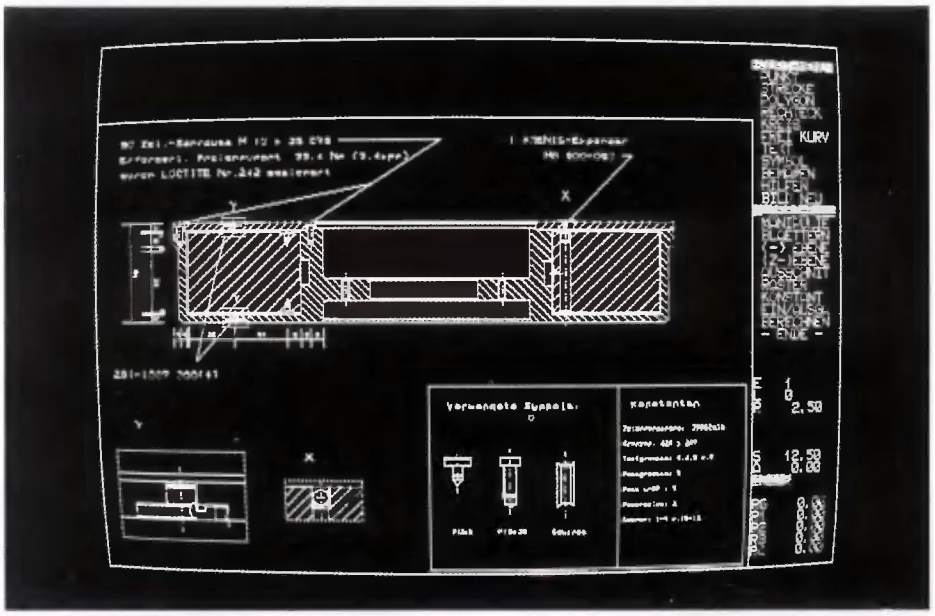
Aby ustalić kontury rzutu pionowego budynku, należy wybrać polecenie *„kreślenie wieloboku"* (niem. Polygonzug). Następnie jest definiowany punkt początkowy odcinka. Drugi koniec odcinka jest przesuwany tak długo, aż osiągnie położenie docelowe. W każdej chwili wyświetlane są długość odcinka oraz kąt jego pochylenia.

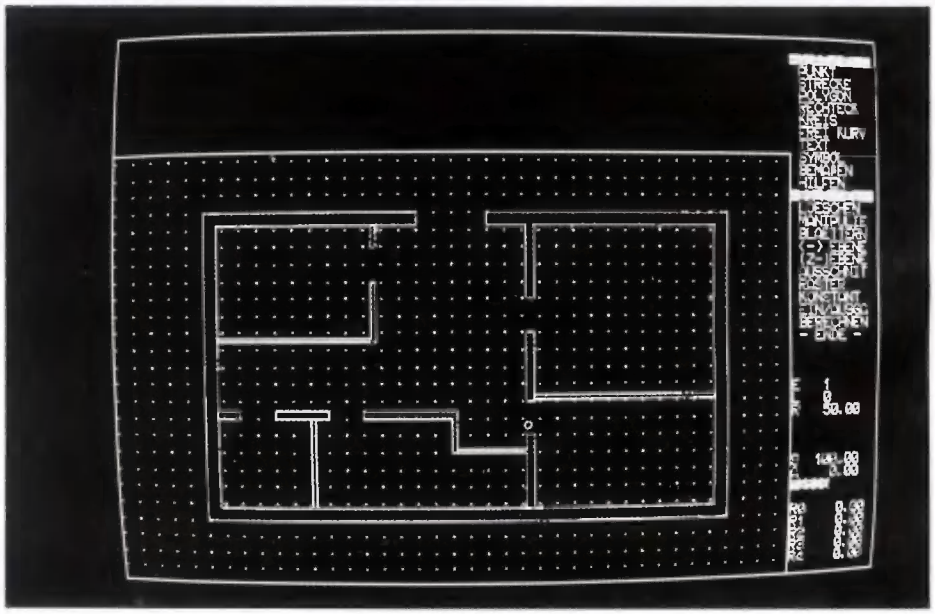
Do dalszego opracowywania szczegółów służy w menu polecenie *„wycinek"* (niem. Ausschnitt). Polecenie podrzędne *„powiększanie"* (niem. Vergrössern) umożliwia np. wybranie za pomocą kursora dowolnego wycinka rysunku w postaci prostokąta. Zawartość tego prostokąta jest w powiększeniu wyświetlana na całym ekranie.

TECHNIKA WARSTW

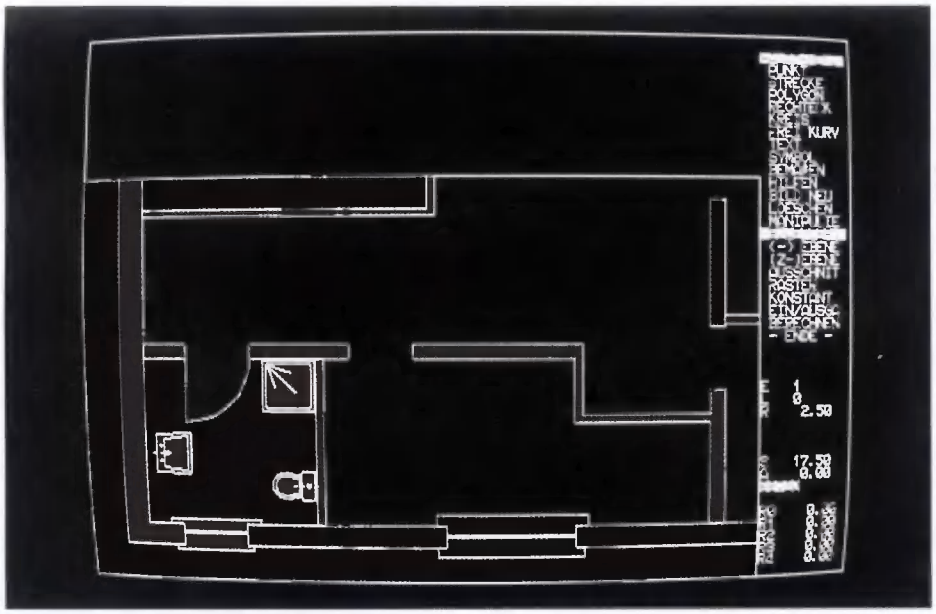
Spośród mnóstwa informacji trzeba wybrać jedynie naprawdę istotne i potrzebne. Umożliwia to zastosowana w programie **Caddy** technika warstw. Poszczególne warstwy działają podobnie jak przezroczyste folie z naniesionymi na nich rysunkami składowymi. Dopiero po nałożeniu jedna na drugą folii widoczny jest kompletny rysunek. Na poszczególnych warstwach można przedstawić, np. instalację elektryczną, ogrzewczą, kreskowanie i wymiarowanie. Każdy z wykonawców poszczególnych rodzajów robót otrzyma tylko rysunek z istotnymi dla niego informacjami, a więc nie będzie musiał zapoznawać się z całością projektu. Program zapewnia możliwość użycia 128 takich warstw.

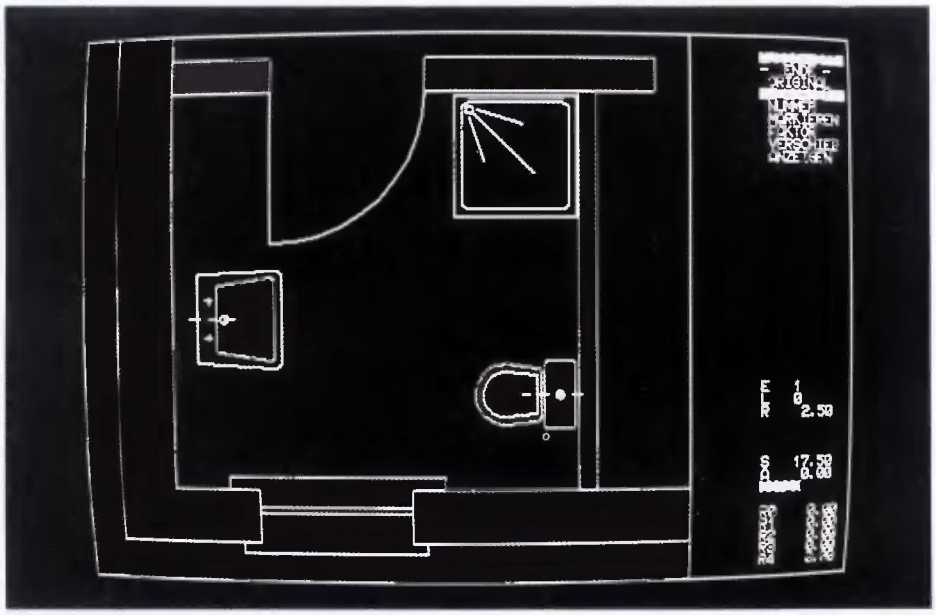


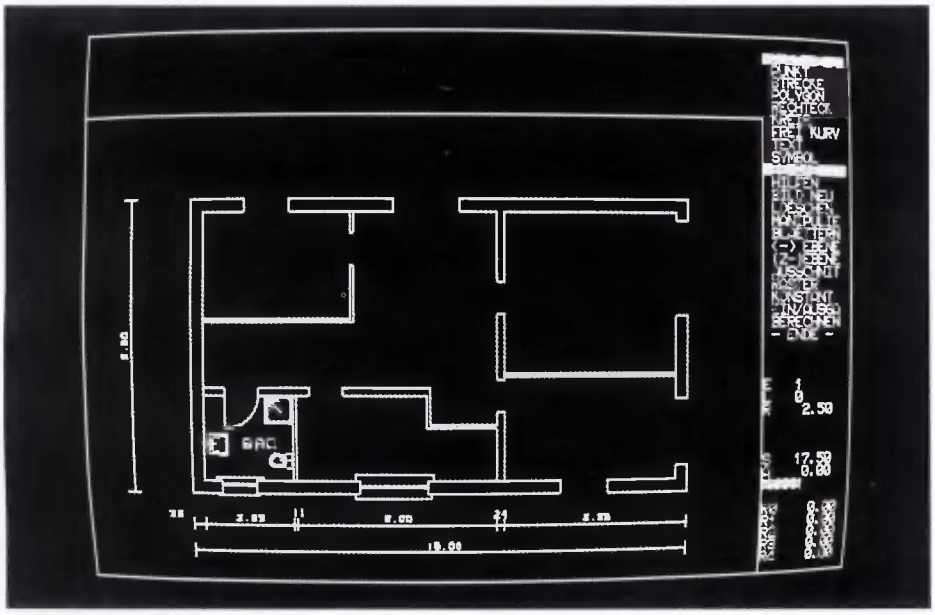




Różne możliwości zastosowania programu Caddy; na dolnym obrazie rzut podstawowy projektu z widocznymi punktami siatki







POMOCE KONSTRUKCYJNE

Użytkownik może korzystać z różnych funkcji pomocniczych. Za pomocą polecenia *„linie równoległe"* (niem. parallele Strecken) łatwo jest np. wykreślić ściany. Polecenie *„odcinanie"* (niem. Abschneiden) działa jak elektroniczna gumka ścierająca, pozwalająca usunąć z rysunku źle umieszczone lub zbędne linie. Równie łatwo na rysunku można wprowadzić w ścianach przerwy okienne oraz drzwi. Użytkownik ma do swej dyspozycji jeszcze wiele innych funkcji ułatwiających pracę.

TECHNIKA SYMBOLI

Jedną z najważniejszych zalet programu jest technika symboli. Pozwala ona po jednorazowym narysowaniu zapamiętać na dyskietce często używane elementy. Symbole te w razie potrzeby są wywoływane przez użycie przyporządkowanej im nazwy i umieszczane w dowolnym miejscu rysunku. Symbolem może być np. WC lub prysznic. Istnieje też możliwość powiększenia elementu, wprowadzania zmian oraz wielokrotnego przedstawiania symboli. Szczególnie wygodne jest też wprowadzanie tekstu opisującego rysunek. Za pomocą kursora można umieszczać znaki pisarskie różnej wielkości

Dotychczas na zwymiarowanie rysunku kreślarka potrzebowała całego popołudnia. Na ekranie praca ta wykonywana jest w ciągu 20 minut, gdyż wystarczy tylko naprowadzić kursor na punkty narożnikowe. Sprawdzenie wymiarów pochodnych nie stanowi również żadnego problemu. Dzięki przezroczystości warstw wymiarowanie przeprowadza się w oddzielnej warstwie.

ROZSZERZENIA

Do systemu można dołączyć ploter o formacie od A4 do A0, a wszelkie zestawienia i teksty są drukowane na drukarkach o różnej wydajności. Systemy **Caddy** można łączyć w sieć lokalną. System może też wymieniać dane z dużym komputerem.

EFEKTYWNOŚĆ

Caddy nie tylko zmniejsza nakład pracy konstruktora. Efektem ubocznym jest znaczne usprawnienie obsługi klientów. Na przykład architekt już po upływie pół godziny jest w stanie zaprezentować klientowi szkic rzutu poziomego zamówionego projektu budynku.

Obecnie wśród potencjalnych użytkowników panuje jeszcze pewien sceptycyzm względem komputeryzacji prac kreślarskich. Dziś jest już jednak oczywiste, że wzrost wydajności pracy inżyniera, a tym samym dotrzymanie kroku konkurencji, są możliwe tylko przez stosowanie projektowania wspomaganego komputerem.

micro

Przykładowe obrazy fragmentu rozwiązania projektowego w różnej skali; na dolnym obrazie po prawej stronie menu główne

Czym jest UNIX

POLECENIA SYSTEMOWE (cd.)

W bieżącym numerze omówiono pozostałe polecenia operowania plikami.

cp

Skoro umiemy już tworzyć nowe pliki, to warto nauczyć się także, jak je najłatwiej kopiować. Niektórzy z nas wiedzą już, jak kopiować pliki używając polecenia cat. Do tego celu służy przede wszystkim polecenie cp (ang. copy), według następującej składni:

cp *plik-wejściowy plik-wyjściowy*

na przykład:

```
$cd ../../..
$cp kos/intel/8086  zal/intel/8086
$
```

co jest najprostszym sposobem zwiększenia stanu posiadania. Jeżeli polubimy tę metodę powiększania swego dorobku, to możemy skopiować jednocześnie większą liczbę plików, przy czym plik wyjściowy musi być skorowidzem, np.:

```
$cp kos/intel/8086  kos/intel/8080 zal/intel
$ls zal/intel
8080
8086
$
```

Sprawdzając zawartość skorowidza *zal/intel* poleceniem **ls** przekonujemy się, że wszystkie kopiowane pliki przepisały się właściwie.

Kopiując pliki poleceniem **cp** należy pamiętać, że poprzednia zawartość pliku wyjściowego ulega zniszczeniu (bez ostrzeżenia). W powyższym przykładzie, drugie wykonanie polecenia **cd** spowodowało powtórne przepisywanie pliku *kos/intel/8086* na już istniejący plik o nazwie *8086* w skorowidzu *zal/intel*, powodując zniszczenie jego poprzedniej zawartości (w tym wypadku, na szczęście, nie poniesiono żadnych strat, gdyż obie zawartości były identyczne).

Oczywiście, przy kopiowaniu plików obowiązują naturalne zasady, że można skopiować tylko plik istniejący, mając prawo do jego odczytu, na istniejący skorowidz lub plik w tym skorowidzu, pod warunkiem posiadania prawa zapisu. W obu wypadkach, tzn. przy nieprawidłowym odczycie lub nieprawidłowym zapisie sygnalizowane są błędy, np. odpowiednio:

```
$cp kos/intel/8085 zal/intel
cp: cannot open kos/intel/8085
$cp kos/intel/8080 zal/intel_80
cp: cannot create zal/intel_80
$
```

Poleceniem **cp** można także wyprowadzić informację, gdy plik wyjściowy będzie odpowiadał urządzeniu wyjściowemu, np.:

```
$cp kos/intel/8080 /dev/tty
$
```

W podobny sposób polecenia **cp** użyć można do utworzenia pliku, co pozostawiamy jako zadanie dla Czytelnika.

mv

Poleceniem **cp** kopiuje się plik wejściowy nie niszcząc go, tak że powstają dwa identyczne jego egzemplarze. Polecenie **mv** działa podobnie, jednak plik wejściowy ulega zniszczeniu, tzn. operacja:
mv *plik-wejściowy plik-wyjściowy*
powoduje nadanie nowej nazwy *plik-wyjściowy* plikowi o nazwie *plik-wejściowy*, tzn. przemianowanie go[1]. Przykładowo, operacja

```
$cd zal/intel
$mv 8086 i8086
$
```

powoduje przemianowanie pliku w obrębie tego samego skorowidza, a operacja:

```
$mv i8086 ../multibus/mb8086
$
```

spowoduje umieszczenie pliku *i8086* w innym skorowidzu, pod inną nazwą, i zniszczenie go w bieżącym skorowidzu.

Podobnie jak poleceniem **cp**, poleceniem **mv** można przesyłać jednocześnie większą liczbę plików, pod warunkiem, że plikiem wyjściowym jest skorowidz.

Typowe błędy popełnione przy operowaniu tym poleceniem polegają na próbie przepisania pliku nieistniejącego lub podjęciu próby przepisania bez posiadania prawa odczytu, np.:

```
$mv 8085 ../multibus/mb8085
mv: cannot access 8085
$
```

Próba przepisania na istniejący plik wyjściowy bez prawa zapisu może zakończyć się powodzeniem, gdyż w tym przypadku nie wyświetla się komunikatu o błędzie, lecz ostrzeżenie z pytaniem, czy mimo takiej sytuacji należy dokonać przepisania.

Nieumiejętne używanie polecenia **mv**, co zdarza się bardzo często, może doprowadzić do spustoszenia systemu plików, gdyż polecenie to usuwa zarówno istniejące pliki wejściowe, jak i wyjściowe. Przykładowo, wykonując kolejno polecenia:

```
$mv ../../schw/8080 intel
$mv ../../schw/cross intel
$
```

spowodujemy przepisanie dwóch plików z zachowaniem ich nazw na skorowidz *zal/intel*. Jednakże, próba wykonania poleceń:

```
$mv intel/8080 ../multibus/mb8086
$mv intel/cross ../multibus/mb8086
$
```

zakończy się tragicznie, ponieważ *mb8086* nie jest nazwą skorowidza, lecz nazwą zwykłego pliku. Tak więc, przepisanie pliku *intel/8080* zniszczy zawartość pliku *mb8086*, a przepisanie pliku *intel/cross* zniszczy zawartość nowego pliku *mb8086*, poprzednio nazywanego *intel/8080*. Zatem, z trzech plików pozostanie nam jeden, nazwany *mb8086* o zawartości *intel/cross*.

rm

Choć dotychczas omówione polecenia umożliwiają niszczenie plików, jest to raczej efekt uboczny ich działania. Poleceniem przeznaczonym specjalnie do tego celu jest **rm** (ang. remove). Przykładowo, można zniszczyć pliki przepisane od innych użytkowników, tj. nie będące naszą własnością, np.:

```
$rm ../multibus/mb8086
$
```

Jeżeli plik jest chroniony przed zapisem, tzn. nie mamy prawa do zniszczenia go, otrzymamy w odpowiedzi ostrzeżenie z pytaniem, czy na pewno chcemy go zniszczyć. Podanie odpowiedzi zaczynającej się od **y** (ang. yes) powoduje zniszczenie pliku. Jednak w niektórych realizacjach systemu UNIX może nie być to możliwe i wtedy otrzymujemy odpowiedź:

rm: file not removed

Posługując się poleceniem **rm** można zniszczyć także skorowidz, lecz w takim wypadku otrzymuje się dodatkową informację o zniszczeniu skorowidza, np.:

```
$rm ../multibus
../multibus: directory
$
```

Szczególnie skuteczne jest używanie tego polecenia ze zmiennymi nazwami plików, np. z nazwą „★ " oznaczającą wszystkie pliki. Należy jednak przestrzec przed nieumiejętnym korzystaniem z tej możliwości, gdyż łatwo jest zniszczyć za dużo, tracąc szanse na późniejsze odtworzenie niepotrzebnie zniszczonych plików.

1) W praktyce, zawartość pliku nie zmienia się w ogóle, zmianie ulegają tylko nazwy plików, które są przenoszone do nowego skorowidza. Wyjątkiem od tej zasady jest przepisywanie plików na inne urządzenie fizyczne, np. z dyskietki na dysk stały, kiedy plik jest przepisywany do innego systemu plików.

INNE OPERACJE DOTYCZĄCE PLIKÓW

Oprócz podstawowych operacji dotyczących plików, jak wyprowadzanie, kopiowanie, przemianowywanie i niszczenie, istnieje grupa poleceń odnoszących się do bardziej specyficznych funkcji plików. Poniżej omówiono polecenia związane ze zmianą informacji o plikach w systemie plików.

chmod

Polecenie **chmod** (ang. change mode) służy do zmiany praw dostępu do pliku. Bity 2-9 określające w całości tryb dostępu ustawia się łącznie jednym poleceniem o składni:

chmod tryb nazwa-pliku

podając jawnie tryb w zapisie ósemkowym według następującej zasady:
– prawa dostępu właściciela $r=2^8=400_8$, $w=2^7=200_8$, $x=2^6=100_8$
– prawa dostępu grupy $r=2^5=40_8$, $w=2^4=20_8$, $x=2^3=10_8$
– prawa dostępu wszystkich $r=2^2=4$, $w=2^1=2$, $x=2^0=1$.
Tak więc, zapewnienia pełnego dostępu dla właściciela, ustanowienia prawa odczytu dla członków grupy oraz zabronienia dostępu wszystkim innym użytkownikom, tzn. rwxr-----, w odniesieniu do pliku *sm4/camac*, można dokonać poleceniem:

$chmod 740 *sm4/camac*
$

Jeżeli podjęto próbę zmiany praw dostępu do nieistniejącego pliku lub do pliku, którego nie jest się właścicielem, to w odpowiedzi pojawi się meldunek o błędzie, np. odpowiednio:

$chmod 740 *sm3/camac*
chmod: cannot access *sm3/camac*
$chmod 740 *../kos*
chmod: cannot change *../kos*
$

Oprócz powyższej metody liczbowego wyrażania trybu dostępu, w niektórych implementacjach systemu UNIX stosuje się metodę symboliczną, której nie będziemy omawiać. Nie będziemy też omawiać rozszerzonych metod ochrony plików, które stosuje na ogół tylko zarządca systemu. Warto jedynie dodać, że wszystkim plikom nadaje się domyślny tryb dostępu w chwili utworzenia – najczęściej jest on równy 666 lub 644 dla zwykłych plików i 777 lub 755 dla skorowidzów.

ln

Jeżeli chcemy używać pliku innego użytkownika lub naszego własnego pliku zapisanego w innym skorowidzu, to nie musimy go kopiować. Wystarczy dołączyć go do naszego skorowidza poleceniem **ln** (ang. link), które powoduje utworzenie łącznika do żądanego pliku. Składnia polecenia jest następująca:
ln *plik nowa-nazwa*

Przykładowo, zamiast kopiować pliki innych członków grupy możemy je dołączyć do naszego skorowidza:

$ln */usr/kos/... i8086*
$ln */usr/schw/...*
$

Należy pamiętać, że polecenie **ln** jest skuteczne tylko w odniesieniu do zwykłych plików. Nie dotyczy skorowidzów, gdyż żaden skorowidz z założenia nie może figurowac w wykazie więcej niż jednego innego skorowidza. Ponadto, nie można dołączyć pliku z innego systemu plików, a więc z innego urządzenia fizycznego.

Warto też wiedziec, że w rzeczywistości po wykonaniu polecenia **rm** zawartość dołączonego pliku nie ulega zniszczeniu, usuwane są jedynie łączniki do pliku. Tak więc, możliwa jest sytuacja, że plik dołączony przez jednego użytkownika, tzn. odnotowany z liczbą łączników równą 2, zniszczony następnie przez oryginalnego właściciela, będzie istniał nadal w skorowidzu nowego użytkownika, lecz z liczbą łączników równą 1. Przykre może być jedynie to, że nie będąc właścicielem pliku, nowy użytkownik nie może zmienić praw dostępu.

mount

W celu dołączenia nowego hierarchicznego systemu plików do już istniejącego, konieczne jest wykonanie polecenia **mount**, tzn. zamontowanie drzewa reprezentującego dołączany system plików w systemie istniejącym. Operacja montowania dotyczy najczęściej wymiennych jednostek pamięci zewnętrznej i polega na umieszczeniu skorowidza pierwotnego (ang. root) dołączanej jednostki w istniejącym skorowidzu jednostki stałej, zamiast innego pustego skorowidza należącego do tego użytkownika, tzn.:

mount */dev/urządzenie skorowidz-zastąpiony*

Przykładowo, zamontowanie dyskietki w skorowidzu może mieć następującą postać:

$mount */dev/df0 /usr/schw*
$

Oczywiście, zarówno plik specjalny, reprezentujący odnośne urządzenie, jak i zastępowany pusty skorowidz muszą istnieć przed wydaniem polecenia **mount**.

W większości realizacji systemu Unix polecenie to jest zastrzeżone dla zarządcy systemu, a sam program znajduje się w skorowidzu *etc*.

umount

Polecenie **umount** (nie unmount) służy do zdemontowania usuwalnej jednostki pamięci zewnętrznej i ma składnię jeszcze prostszą od polecenia **mount**:

umount */dev/urządzenie*

Przykładowo, wydanie polecenia:

$umount */dev/df0*
$

oznacza odłączenie dyskietki uprzednio dołączonej poleceniem **mount**. Skorowidz zawierający uprzednio skorowidz pierwotny tej jednostki staje się pusty.

★ ★ ★

Istnieje więcej poleceń dotyczących operowania plikami, lecz nie są one tak istotne dla początkujących użytkowników. Dlatego pominiemy ich szczegółowe omówienie, poprzestając na wyliczeniu niektórych z nich:

file – podaje typ danych zawartych w pliku, tj. rodzaj pliku (pusty, wykonywalny, skorowidz, dane liczbowe, polecenia, tekst ASCII, tekst angielski, tekst programu w języku C, Fortran itp.)

find – służy do hierarchicznego przeszukiwania całego systemu plików w celu odznalezienia pliku mającego określone właściwości, np. nazwę, właściciela, prawa dostępu, rozmiar, datownik itp.

tail – powoduje wyprowadzenie kilku ostatnich wierszy pliku

wc (ang. word count) – określa liczbę słów, wierszy i znaków w pliku.

Polecenia i operacje dotyczące plików specjalnych, tj. reprezentujących urządzenia zewnętrzne, są równie ważne jak dotąd omówione, lecz mają nieco inną naturę ze względu na odmienność plików specjalnych od plików zwykłych i skorowidzów. Z tego względu poprzestaniemy na wyliczeniu tych poleceń, odkładając ich omówienie na inną okazję. Są to głównie operacje związane z wprowadzeniem informacji, jak:

pr (ang. print) – formatowanie i drukowanie tekstu

lpr (ang. line printer) – formatowanie i drukowanie tekstu w tle (ang. background), czyli tzw. spooling

tar (ang. tape archive) – kopiowanie systemu plików wraz z zawartością na taśmę magnetyczną i z taśmy

tee (ang. T-junction, „teownik") – wyprowadzenie zawartości pliku jednocześnie na dwa wyjścia (np. plik zwykły i terminal).

JANUSZ ZALEWSKI

Co to jest CAD/CAM?

Historia przetwarzania danych graficznych (ang. computer graphics) sięga wczesnych lat sześćdziesiątych. Wprowadzono wówczas terminale ekranowe umożliwiające konwersację użytkownika z komputerem. Pionierami w tej dziedzinie były: przemysł lotniczy i motoryzacyjny. Komputer w połączeniu z terminalem ekranowym wykorzystywano do projektowania i konstruowania. Przez długi okres na przetwarzanie danych graficznych mogły sobie pozwolić jedynie duże przedsiębiorstwa, realizujące złożone zadania konstrukcyjne. Spowodowane to było bardzo wysokimi wówczas kosztami procesorów, pamięci i terminali. Dopiero niedawno, w konsekwencji stałej obniżki cen oraz coraz lepszych parametrów mikroprocesorów i pamięci, dyskusja na temat przetwarzania danych graficznych znów się ożywiła.

NOWE DZIEDZINY ZASTOSOWAŃ

Z upływem czasu poszerzył się zakres zastosowań przetwarzania danych graficznych. Grafikę stosuje się już nie tylko przy projektowaniu wspomaganym komputerem. Zastosowania można podzielić na cztery grupy:

- graficzne przedstawianie danych alfanumerycznych,
- interaktywne planowanie, projektowanie i konstruowanie,
- analiza obrazu,
- graficzne przetwarzanie danych powiązane z przetwarzaniem tekstów.

Przedstawianie danych alfanumerycznych w formie graficznej jest stosowane w wielu gałęziach techniki i gospodarki. Dzięki grafice można lepiej przedstawić i jednocześnie szybciej ogarnąć i zrozumieć złożone zależności. Inżynier otrzymuje wyniki pomiarów i obliczeń w formie najwygodniejszej do dalszej analizy. Handlowiec zamiast dużej liczby danych w formie list i kolumn znaków otrzymuje przejrzyste diagramy. Użytkownicy mogą uzyskać wyniki w postaci różnorodnych diagramów: słupkowych, kołowych, czasowych, przepływowych itp.

PROJEKTOWANIE WSPOMAGANE KOMPUTEREM

Projektowanie wspomagane komputerem (ang. Computer Aided Design – CAD), to przede wszystkim zastosowanie komputera do pracy projektowej. Obecnie zalicza się do niego także przetwarzanie danych geometrycznych (czyli graficznych) oraz optymalizację wykrojów.

Wytwarzanie wspomagane komputerem (ang. Computer Aided Manufacturing – CAM), to stosowanie komputera w procesach przygotowania produkcji i wytwórczym, jak również mające już piętnastoletnią historię tworzenia programów do obrabiarek sterowanych numerycznie (ang. NC – numerical control).

CAD obejmuje konstruowanie części mechanicznych. Projektant zastępuje deskę kreślarską ekranem monitora. Konstruowanie nie oznacza jedynie wykonywania rysunków na desce kreślarskiej, lecz także obliczenia, sprawdzanie i dokumentowanie. Komputer wspomaga te prace i zapamiętuje istotne informacje.

Ocenia się, że praca przy desce w czasie formułowania zadania zajmuje konstruktorowi od 20 do 40% całkowitego czasu pracy. Dzięki zastosowaniu CAD, czas pracy na tym wstępnym etapie można skrócić nawet o 75%.

System graficzny już dzisiaj odciąża konstruktora przy rutynowych, mało twórczych czynnościach. System kopiuje rysunki lub wyprowadza je w zmienionej skali, odwraca lub przesuwa elementy graficzne oraz wykonuje wymiarowanie, opisywanie i usuwanie fragmentów rysunku.

Praca konstruktora tylko w nielicznych wypadkach polega na tworzeniu nowatorskich konstrukcji. Dlatego sporą część pracy konstruktora przy ekranie monitora stanowi dziś „projektowanie wariantowe".

Większość podstawowych konstrukcji części lub zespołów, jest znana i zapamiętana w systemie, skąd są one wybierane zgodnie z potrzebami wykonywanego zadania.

W praktyce wygląda to tak: komputer proponuje konstruktorowi wstępny projekt określonej części, wybrany spośród zapamiętanych podstawowych konstrukcji geometrycznych. Konstruktor dostosowuje go do postawionych wymagań, zmieniając odpowiednio różne wymiary.

Programy projektowania wariantowego mogą być bardzo efektywne. Cały proces projektowania, często nawet łącznie z przygotowaniem oprzyrządowania urządzeń wytwórczych, jest wspomagany programami uwzględniającymi konkretne warunki techniczne.

INTERAKTYWNA PRACA I DOSTOSOWANIE DO POTRZEB UŻYTKOWNIKA

Rzeczywiście nowatorskie konstruowanie, a więc projektowanie i opracowywanie nowych części lub zespołów, wymaga najczęściej dwuwymiarowego, a czasami nawet trójwymiarowego przedstawienia i rzutowania części.

Narzuca to pewne wymagania na oprogramowanie wspomagające system graficzny, który musi działać interaktywnie i w sposób dostosowany do potrzeb użytkownika.

Przy przygotowywaniu produkcji można korzystać z systemu CAD na różne sposoby. Ponieważ narzędzie czy forma jest często negatywem wytwarzanej części, inżynier przygotowujący produkcję może zaczerpnąć dane opisujące geometrię części z systemu CAD, a następnie odpowiednio je opracować. Podobnie programista, przygotowujący programy dla obrabiarek sterowanych numerycznie, może korzystać z zapamiętanych już informacji o danej części.

PROJEKTOWANIE UKŁADÓW SCALONYCH I MAP

Optymalizacja wykrojów jest znanym zadaniem, na przykładzie którego szczególnie dobrze można zademonstrować efektywność elektronicznego przetwarzania danych. Dzięki takiej optymalizacji można zaoszczędzić milionowe kwoty. Na przykład, do wycinania grubszych blach coraz częściej stosowane są palniki sterowane numerycznie. Ekonomiczne wykorzystanie tych maszyn wymaga pocięcia za jednym razem całego arkusza blachy. Jeśli poszczególne części opracowywanej konstrukcji znajdują się już w pamięci komputera, mogą być one bezpośrednio użyte do przeprowadzenia optymalizacji.

Wielkie znaczenie ma dziś przetwarzanie danych przy projektowaniu rozłożenia elementów w półprzewodnikowych układach o wielkim stopniu scalenia. Układy te są niezwykle skomplikowane, a na każdej płytce krzemu, o boku długości kilku milimetrów, trzeba zmieścić wieleset tysięcy elementów przełączających. Zastosowanie komputera jest w tym wypadku niezbędne.

Dziedziną zastosowań o nie w pełni jeszcze ustalonych granicach zastosowań jest przetwarzanie danych geograficzno-geometrycznych (ang. mapping). Danymi wejściowymi są wyniki pomiarów geometrycznych powierzchni, które należy zobrazować lub opracować. Zmierzone współrzędne są wprowadzane do komputera i mogą być przedstawione graficznie na mapie. Można w

Przegląd oprogramowania CAD dla mikrokomputerów

Dostawca	Oprogramowanie	Liczba wymiarów	Mikrokomputer	System operacyjny	Minimalna pojemność pamięci	Dyskietki lub dyski sztywne	Wyprowadzanie	Sprzęg programowy	Sprzęg do sterowania numerycznego	Obliczenia	Biblioteka	Liczba warstw	Zmiana skali (200 m)	Wymiarowanie	Kreskowanie	Wprowadzanie	Zastosowanie
EOPG (A)	AutoCAD	2	Victor9000 /Sirius1 IBM PC Z100 NEC APC Rainbow NCR MateV DUET16	MS DOS1.25 PC DOS1.1/2.0/2.11 MS DOS1.01/1.25 CP/M86 MS DOS 2.05 MS DOS MS DOS	384 256 256 256 256 256 256	fl	pl	■	▲	■	■	127	★	aut	▲	kl, ms dig, lp	rysunki dwuwymiarowe w architekturze, instal., elektr. itp.
Folger Geretsegger	AutoCAD	2	Z100	MS DOS1.01.1.25	256	2 fl, hd	pl	★	–	★	★	127	■	aut	▲	kl,ms, dig, lp	rysunki dwuwymiarowe w architekturze, instal. itp.
Folger Geretsegger	DASOFT	2	Z100	CPM2.2	128	2 fl	pl	★	–	★	gdu	3	★	–	–	kl, dig	projektowanie płytek drukowanych wielowarstwowych
infoCommerz	Superdraft	2	IBM PC	PC DOS2	256	1 fl, hd	pl, dr	▲	–	–	tz	64	▲	aut	▲	kl, dig	kreślenie dwuwymiarowe
Kontron	AutoCAD	2	ErgoPC	MS DOS2.11	256	1 fl, hd	pl	■	–	■	gdu	127	★	aut	■	kl, dig	rysunki dwuwymiarowe w architekturze, elektronice itp.
Kontron	Laygraph	2	PSI98QW20	KOS	256	1 fl	pl, dr	–	–	–	tz		★	man	–	kl, dig	kreślenie dwuwymiarowe
Olivetti (A)	AutoCAD	2	M24	MS DOS2.11	128	2 fl, hd	pl	★	–	■	gdu	127	★	aut	★	kl, ms, dig, lp	kreślenie dwuwymiarowe
Pero (A)	Perograf2D	2	Sirius1, Vicki, TI-PC, Apricot	MS DOS PC DOS	256	fl,hd	pl,dr	■	–	■		78	■	aut + man	★	kl, dig,lp	dwuwymiarowe przedstawianie danych pomiarowych
RSE	MiniCADO	2	IBM PC	MS DOS.2.0,CP/M86	256	1 fl,hd	pl	★	–	–	tz	2	★	aut	★	kl,dig	kreślenie dwuwymiarowe
Typolt	Statik	3	IBM PC	MS DOS,CP/M86	64	hd	pl,dr	–	–	★	–		–		■	kl	analizy statyczne
Victor (A)	AutoCAD	2	Sirius1	CP/M86, MS DOS 1.25,2.11	384	2 fl	pl	★	–	■	gdu	127	★	aut	★	kl,ms, dig, lp	kreślenie dwuwymiarowe
Zema (A)	3D-Graphics	3	Twin, Apple II	DOS3.3	48	1 fl	pl,dr	–	–	–			■	★	★	kl	grafika trójwymiarowa, mała rozdzielczość
Zema (A)	Platinen-Layout	2	Twin, Apple II	DOS3.3	48	1 fl	dr	–	–	★	■	2	★	■		kl	projektowanie płytek drukowanych
Zema (A)	PitStik	2	Twin, Apple II	DOS3.3	64	1 fl	pl	■	–	–	gdu		★		–	dig	interaktywny system graficzny
Zema (A)	Baustatik-Pakete	2	Twin, Apple II	DOS3.3	64	1 fl	pl,dr	–	–	■	★					kl	analiza i projektowanie elementów statycznych
Pero (A)	PERODESIGN	3	Sirius1, Vicki, Apricot, TI-PC, NCR MateV, Nixdorf PC	MS DOS PC DOS	256	1 fl,hd	pl,dr	★	–	★	gdu	130	■	aut	■	kl,ms, dig,lp	trójwymiarowy pakiet do planowania mieszkań z automatycznym sporządzaniem oferty i zamówień, automatyczne sporządzanie planów inwestycyjnych

LEGENDA:
A – autoryzowany sprzedawca, aut – automatyczne, dig – digitajzer, dr – drukarka, fl – dyskietki, gdu – gotowa do użycia, hd – dyski sztywne, kl – klawiatura, lp – pióro świetlne, man – ręczne, ms – myszka, pl – ploter, tz –trzeba założyć

ten sposób przedstawić sieci kolejowe, elektryczne czy rurociągowe rozciągające się na dużych obszarach. Opracowania takie potrzebne są także w przemyśle, przy projektowaniu fabryk, prowadzeniu instalacji elektrycznych lub pneumatycznych.

SIATKA O KWADRATOWYCH OCZKACH

Analiza (graficzne przetwarzanie) obrazów najczęściej dotyczy zdjęć fotograficznych. Do opracowania takiego obrazu przez komputer niezbędna jest najpierw jego digitalizacja. W tym celu na obraz jest nakładana jakby siatka, zaś każdemu jej kwadratowi (elementowi obrazu) przyporządkowuje się wartość numeryczną będącą miarą intensywności (jasności) elementu. Za pomocą tych wartości można przedstawić cyfrowo stopnie szarości lub rodzaje barwy. Digitalizacji obrazów dokonuje się za pomocą specjalnych urządzeń próbkujących, zwanych skanerami, w których promień świetlny omiata obraz punkt po punkcie (podobnie jak w kamerze telewizyjnej).

GRAFIKA Z TEKSTEM

Przy przetwarzaniu danych graficznych na ogół nie stosuje się przetwarzania tekstów. Są jednak zastosowania, np. wytwarzanie katalogów i cenników, w których równocześnie z rysunkami i obrazami niezbędne jest przedstawianie tekstu różnej wielkości i w dodatku pisanego różną czcionką. Wtedy system graficzny okazuje się niezbędny. Pomocnicze środki techniczne, np. wymienne generatory znaków oraz różne możliwości rysunków perspektywicznych pozwalają na interaktywne łączenie tekstu, grafiki i ilustracji w formę przeznaczoną do wydrukowania.

Jeśli przedsiębiorstwo zamierza zastosować wiele monitorów graficznych z elastycznym dostępem do bazy danych, to ekonomicznie uzasadnione będzie użycie komputera centralnego z rozszerzalną pamięcią. Praktyka wykazała bowiem, że bazy danych powstałe z przetworzenia danych graficznych zwiększają swą objętość w stosunkowo krótkim czasie.

WYMIANA INFORMACJI MIĘDZY SYSTEMAMI GRAFICZNYMI

Istotną sprawą jest wspomagane przez komputer wyszukiwanie takich informacji, jak normy i właściwości materiałów. Warto też pamiętać, że w wielu pracach projektowych zastosowanie systemów graficznych poprawia jakość i dokładność konstrukcji oraz zmniejsza liczbę błędów.

W większości wypadków okazało się, że wprowadzenie systemów CAD w działach konstrukcyjnych jest ekonomicznie opłacalne. Aby systemy graficzne przyniosły zysk dla całego zakładu, dane przetworzone w jednym dziale powinny być udostępniane również innym działom.

micro

Komputer osobisty DIALOG DTC-8

Prezentowany poniżej system stanowi kolejną propozycję sprzętu mikrokomputerowego spoza przemysłu kluczowego. Rozszerza on możliwość zaspokojenia potrzeb tych wszystkich, którzy komputer pragną wykorzystać jako narzędzie w biurze, na stanowisku pomiarowym lub w hali produkcyjnej. Zapotrzebowanie na takie właśnie systemy jest ogromne, a ich podaż w kraju – nadal niedostateczna. Mamy nadzieję, że DIALOG wypełni, chociaż częściowo istniejącą lukę. Zainteresowanych odsyłamy do autora (tel. 55-24-24, Warszawa).

W czasie, gdy na świecie spada zainteresowanie komputerami osobistymi firmy IBM, gdy nowatorski Macintosch doczekał się wreszcie bogatego oprogramowania, a Amerykanie gorączkowo oczekują pojawienia się w sklepach mikrokomputera Amiga, w Polsce wzrasta popularność maszyn IBM PC. Pewna liczba tych urządzeń dotarła już do rąk krajowych użytkowników, dzięki ofercie działających w Polsce przedsiębiorstw zagranicznych. Moda na IBM PC wytworzyła się głównie za sprawą dopływu dużej ilości oprogramowania użytkowego. Rozpowszechnił się nawet pogląd, że „prawdziwy" komputer osobisty (ang. personal computer) musi mieć mikroprocesor co najmniej 16-bitowy. Uważa się, że epoka mikroprocesorów 8-bitowych już przeminęła i spotyka się je jeszcze jedynie w komputerach domowych (ang. home computers).

Tymczasem o walorach użytkowych mikrokomputera, a w szczególności o jego przydatności do praktycznych zastosowań, nie decyduje wcale długość słowa mikroprocesora. O wiele większe znaczenie mają inne cechy konstrukcyjne: jakość „interfejsu z operatorem" (monitor ekranowy, a raczej jego sterownik i klawiatura oraz ich obsługa przez oprogramowanie systemowe), szybkość i niezawodność pamięci dyskowych, dostępność oprogramowania użytkowego. Udowodniła to niedawno firma Amstrad, wprowadzając na rynek rodzinę mikrokomputerów osobistych Amstrad CPC zawierających mikroprocesor Z80 A. Mikrokomoputery te w krótkim czasie zdobyły rynki europejskie, opanowane przecież przez wyroby takich firm jak Commodore, Atari czy Sinclair. Stało się to możliwe dzięki temu, że maszyny Amstrad zostały wyposażone w cechy niezbędne dla typowych, praktycznych zastosowań redagowania tekstów, tworzenia baz danych czy kalkulacji tabelarycznych – w odróżnieniu od dominujących dotąd gier czy programów edukacyjnych. Najważniejsze z tych cech to: możliwość wyświetlania tekstu w 80 kolumnach i obrazów graficznych o dużej rozdzielczości oraz szeroko rozpowszechniony, uniwersalny, dyskowy system operacyjny (w wypadku komputerów Amstrad jest nim CP/M). Własności te natychmiast udostępniły posiadaczom maszyn Amstrad CPC istniejące, bogate oprogramowanie użytkowe, z programem WordStar oraz dBase II na czele.

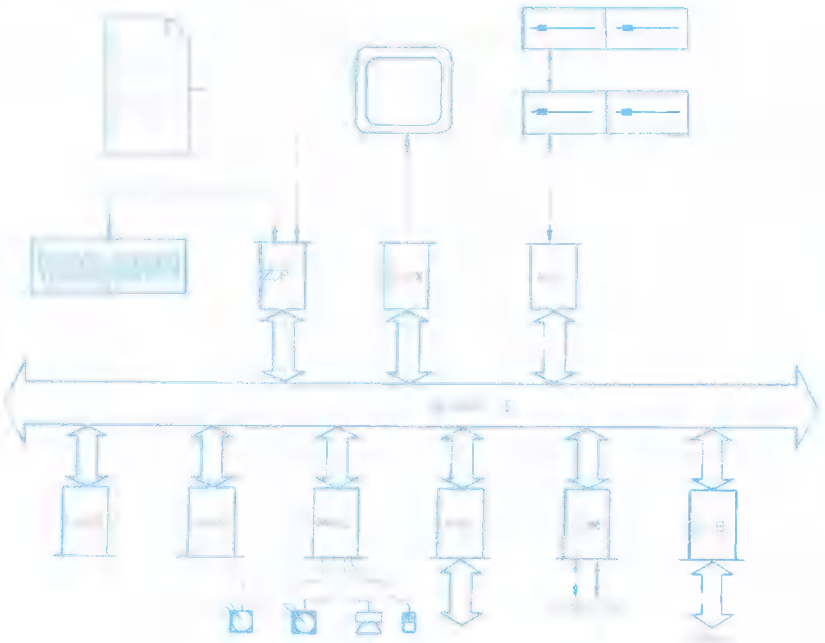
Zdając sobie sprawę z dużej konkurencji na rynku komputerów osobistych, podczas projektowania DTC-8 największy nacisk położono na zapewnienie użytkownikowi jak największej wygody. W wyniku analizy słabych stron powszechnie znanych mikrokomputerów powstała poniższa lista własności, jakie powinien mieć DTC-8:

- modularność – uniwersalna szyna zapewniająca łatwość rozbudowy przez producenta, inne firmy, a także przez użytkownika,
- klawiatura, wyposażona w programowane klawisze funkcyjne oraz klawisze sterowania kursorem w układzie „kierunkowym", położone w miejscu łatwo dostępnym z zasadniczej części klawiatury,
- sterownik monitora, umożliwiający wyświetlanie tekstu w 80 kolumnach, a także obrazów graficznych z niezależnym dostępem do indywidualnych punktów ekranu, z tym, że pamięć obrazu powinna znajdować się poza przestrzenią adresową procesora,
- popularny dyskowy system operacyjny.

DTC-8 składa się z trzech elementów: modułu podstawowego, klawiatury i monitora

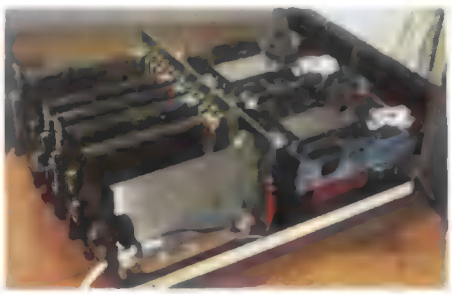
Wydaje się, że w procesie projektowania DTC-8 udało się spełnić wszystkie wymienione wymagania, a także wprowadzić wiele dodatkowych cech wyróżniających go spośród innych konstrukcji tej klasy.

mikroklan



Schemat blokowy mikrokomputera DIALOG DTC-8

Moduł sterownika graficznego DMC-GDCX



Wnętrze prototypu DTC-8

Schemat blokowy organizacji komputera DTC-8 jest przedstawiony na rysunku. Poniżej opisano jego najważniejsze bloki funkcjonalne.

MAGISTRALA

Aby zapewnić wysoki stopień modularności, należało zastosować uniwersalną, równoległą magistralę międzymodułową. Wybór sygnałów magistrali, a także konstrukcja mechaniczna modułów mają istotne znaczenie dla rozwoju zastosowań systemu.

Kiedy w 1981 roku wraz z Andrzejem Matysem i Leszkiem Zdawskim przystępowaliśmy do definiowania uniwersalnej magistrali dla modułów mikrokomputerowych, sformułowaliśmy dla niej następujące wymagania:

- szyna adresowa 20-bitowa (czyli przestrzeń adresowa pamięci 1 MB), zapewniająca perspektywę zastosowania mikroprocesorów 16-bitowych serii 8086/8088,
- minimum 8 linii przerwań oraz przerwanie wyprzedzające zanik zasilania,
- możliwość pracy wieloprocesorowej,
- złącza pośrednie (bardziej niezawodne i łatwiejsze w użyciu niż krawędziowe),
- standard mechaniczny pojedynczej Eurokarty (dobrze zdefiniowany i rozpowszechniony, niewielkie wymiary pakietów – 100×160 mm, umożliwiające zestawianie „poręcznych" systemów).

Żaden z rozpowszechnionych wówczas na świecie standardów nie spełniał powyższych wymagań. Dlatego zdecydowaliśmy się na opracowanie własnego interfejsu międzypakietowego. Został on opatrzony nazwą BUSMAT i zastosowany w Modułowym Systemie Mikroprocesorowym MSM, produkowanym do dziś przez firmę „Impol 1". Złącze 64 – stykowe typu Cannon, zastosowane do łączenia pakietów z magistralą jest produkowane w Polsce. Format pakietów jest na tyle mały, że pozwala budować urządzenia łatwo przenosne, a jednoczesnie dostatecznie duży, aby przy obecnej technologii układów scalonych zmiescic na oddzielnych płytkach funkcjonalnie zamknięte bloki mikrokomputerowe.

Od czasu powstania BUSMAT-u pojawiło się na swiecie kilka nowoczesnych standardów magistrali i to właśnie dla formatu pojedynczej Eurokarty (VME, MULTIBUS II[1]). Jednak sensowne ich wykorzystanie wymaga zastosowania drogich i trudno dostępnych, specjalizowanych układów scalonych. Z drugiej strony BUSMAT okazał się bardzo wygodny i uniwersalny, a ponadto już kilkaset systemów wykorzystujących ten standard znajduje się w rękach użytkowników.

W takiej sytuacji naturalne stało się przyjęcie standardu BUSMAT w mikrokomputerach DIALOG. Dokonana została jednakże pewna modyfikacja. Polega ona na odizolowaniu jednej linii zasilającej +5 V i przeznaczenia jej do odprowadzania napięcia doładowującego akumulatory, umieszczone na niektórych pakietach. Tak zmodyfikowany standard interfejsu międzypakietowego, nazwany BUSMAT II, został zastosowany w mikrokomputerze DIALOG DTC-8.

Na sygnały BUSMAT II składają się napięcia zasilające +5 V, ± 12 V, ± 15 V, oddzielone napięcie akumulatorowe +5 BAT, 8 linii danych, 20 linii adresowych, 10 linii przerwań oraz linie sygnałów sterujących. BUSMAT II pozwala na adresację 1MB pamięci, pracę wieloprocesorową oraz obsługę awarii zasilania sieciowego (przerwanie uprzedzające zanik napięc zasilania, oddzielne napięcie +5 V BAT).

Należy dodać, że zastosowanie modułu systemu MSM (BUSMAT) w systemie DMC lub w komputerze DTC-8 (BUSMAT II) wymaga przecięcia jednego połączenia na module MSM. Zastosowanie modułu DMC w systemie MSM nie wymaga żadnych modyfikacji.

KONSTRUKCJA MECHANICZNA

Większość komputerów osobistych zbudowanych jest na jednym dużym obwodzie drukowanym zawierającym procesor, układy pomocnicze oraz ewentualnie pamięć (IBM PC, Apple II). Ta podstawowa płyta (ang. motherboard) ma złącza do montowania modułów z układami obsługi urządzeń we-wy, z dodatkową pamięcią itp.

Przy realizacji DTC-8 przyjęto inne rozwiązanie. Wszystkie moduły z procesorem włącznie rozmieszczono na jednorodnych płytkach-pakietach. Rozwiązanie to ma swoje wady i zalety. Do wad należy zaliczyć podwyższony koszt wersji minimalnej (bez modułów dodatkowych). Wśród zalet można wymienić większą elastyczność systemu, możliwość zastosowania chłodzenia konwekcyjnego, uproszczenie konstrukcji mechanicznej.

DTC-8 składa się z trzech elementów łączonych kablami: modułu podstawowego, klawiatury i monitora ekranowego. Moduł podstawowy ma kasetę z magistralą systemową, zawierającą złącza dla 17 pakietów. Pakiety wsuwa się od tyłu bez konieczności rozbierania obudowy. Dwie stacje dysków elastycznych 5,25" umieszczone są poziomo obok siebie w przedniej części obudowy. Zasilacz znajduje się pod stacjami dysków. Takie rozmieszczenie zespołów pozwoliło uzyskać korzystne, ze względu na małą szerokość, proporcje wymiarów: szerokość 320 mm, głębokość 420 mm i wysokość 150 mm.

[1] W standardach tych stosowane są również pakiety o formatach podwojnej Eurokarty.

KLAWIATURA

W układzie klawiatury zastosowany jest mikrokomputer jednoukładowy (ang. single-chip microcomputer) typu 8035 z zewnętrzną pamięcią programu 2716. Wszystkie funkcje klawiatury, takie jak przeszukiwanie matrycy klawiszy, obsługa wielokrotnych wciśnięć, automatyczne powielanie kodu trzymanego klawisza (ang. auto repeat), sterowanie diodami świecącymi, kodowanie i komunikacja z komputerem, są realizowane programowo przez wspomniany układ 8035.

Klawiatura ma 96 klawiszy, łącznie z klawiszami sterowania kursorem, wydzielonymi klawiszami numerycznymi oraz dziesięcioma klawiszami funkcyjnymi. Trzy diody świecące sygnalizują tryb pracy klawiatury.

MODUŁY

Wszystkie układy wchodzące w skład podstawowej konfiguracji DTC-8 zostały rozłożone na trzy pakiety: procesora DMC-ZCPU, sterownika monitora DMC-GDCX oraz sterownika pamięci dyskowych DMC-FDC. Najważniejsze układy pomocnicze to:

- pakiet komunikacyjny DMC-COM, zawierający dwa niezależne kanały interfejsu szeregowego RS232C z kompletem sygnałów dla modemów i programowanym generatorem prędkości transmisji (z własnym rezonatorem kwarcowym),

- pakiet pamięci RAM z zasilaniem bateryjnym DMC-NVMB, pracującej jako dodatkowa pamięć operacyjna lub jako dysk elektroniczny dla zapisu i odczytu,

- pakiet pamięci ROM/EPROM DMC-ROM pracującej jako dysk elektroniczny dla odczytu,

- pakiet uniwersalnych wejść-wyjść równoległych DMC-PIB, zawierający 4 porty 8-bitowe i 2 porty 1-bitowe z programowanym kierunkiem transmisji,

- pakiet wielofunkcyjny DMC-MFC, zawierający zegar-kalendarz zasilany akumulatorowo, programowany układ trzech liczników-dzielników częstotliwości, 3-kanałowy układ generacji dźwięków oraz wejścia dla dwóch drążków sterowniczych (ang. joystick),

- pakiet sterownika szyny GPIB[2] (General Purpose Interface Bus) DMC-GPIB.

Procesor

Moduł procesora DMC-ZCPU z mikroprocesorem Z80B, pracujący z zegarem 5,5 MHz, ma układ zarządzania pamięcią (ang. MMU – Memory Management Unit), który umożliwia przepisywanie zawartości pamięci pomiędzy bankami (każdy bank ma wielkość 64 KB; 16 takich banków można umieścić na magistrali i obsłużyć przez moduł procesora). Można również przekazywać sterowanie do innych banków pamięci – procesor może wykonywać program w dowolnym banku. Mechanizm taki umożliwia stworzenie oprogramowania wykorzystującego dodatkowe obszary pamięci jako szybki i niezawodny dysk elektroniczny. Ponadto DMC-ZCPU zawiera: 64 KB pamięci dynamicznej RAM; do 32 KB pamięci EPROM, która może być programowo włączana lub wyłączana do/z przestrzeni adresowej w dowolnym banku (ang. shadow ROM); układ wejść-wyjść z układem scalonym 8255A, który obsługuje klawiaturę i drukarkę, a może być również wykorzystany do szybkiej komunikacji między procesorami w systemach wieloprocesorowych oraz do dołączenia innych urządzeń; układ przerwań z układem scalonym 8259A; układ zegara generujący przerwania co 1000, 100 lub 20 ms, wykorzystywany do pomiaru czasu lub do odmierzania interwałów czasowych w systemach wielozadaniowych.

STEROWNIK MONITORA EKRANOWEGO

Głównym elementem sterownika monitora DMC-GDCX jest mikroprocesor graficzny firmy NEC – µPD 7220. Ponadto pakiet ten zawiera układ do płynnego przesuwania obrazu w kierunku poziomym (ang. smooth horizontal scrolling), układ do powiększania obrazu (ang. zoom), rejestr do generowania dźwięku oraz własną pamięć obrazu o pojemności 128 KB. Pamięć ta jest odizolowana od przestrzeni adresowej procesora głównego, a dostęp do niej jest realizowany przez rejestry oraz bufor FIFO (First In – First Out) mikroprocesora graficznego. Możliwe jest zastosowanie dwóch lub więcej modułów DMC-GDCX w jednym systemie w trybie synchronicznym dla uzyskania grafiki barwnej. Parametry obrazu: grafika – 640 × 256 punktów, test – 80 kolumn, 25 wierszy oraz dodatkowo trzy wiersze przeznaczone dla wyświetlania opisów klawiszy funkcyjnych i statusu systemu. Sterownik umożliwia płynne przewijanie zawartości ekranu w kierunku poziomym i pionowym oraz mieszanie tekstu i obrazów graficznych bez konieczności zmiany trybu jego pracy.

Sterownik pamięci dyskowych

Moduł syterownika dysków DMC-FDC, zawiera układ scalony µPD 765 oraz dodatkowe układy pomocnicze, które pozwalają dołączyć cztery napędy dyskowe 8", 5,25" lub 3,5" z możliwością tworzenia konfiguracji mieszanych tych napędów. Zapis jednostronny, lub dwustronny, z pojedynczą lub podwójną gęstością.

OPROGRAMOWANIE

Podstawowy system operacyjny komputera DTC-8, DS-DOS, jest zgodny z systemem CP/M. Program BIOS (Basic Input/Output System), poza standardowymi procedurami obsługi dysków i konsoli, zawiera rozszerzenia, umożliwiające wygodne posługiwanie się interfejsem szeregowym i równoległym, klawiaturą i słownikiem graficznym bez konieczności wnikania w szczegóły rozwiązań sprzętowych. Procedury obsługi konsoli realizują również funkcje graficzne, rekonfigurację klawiatury, zmianę trybu wyświetlania tekstu itp. za pomocą sekwencji sterujących zaczynających się od znaku ESC. W procedurach obsługi dysków uwzględniono użycie dodatkowej pamięci RAM i EPROM (moduły DMC-NVMB, DMC-ROM) jako tzw. dysków elektronicznych.

Dostępne jest również oprogramowanie narzędziowe: przemieszczalne makroasemblery, programy wspomagające uruchamianie, programy do formatowania i kopiowania dysków, kompilatory i interpretery języków wysokiego poziomu itp. Bogate oprogramowanie użytkowe obejmuje systemy baz danych, programy do redagowania tekstów, wykonywania kalkulacji ekonomicznych i obliczeń inżynierskich oraz naukowych.

KIERUNKI ROZWOJU SYSTEMU

Opracowywane są dalsze moduły rodziny DMC zwiększające obszar zastosowań komputera DTC-8. Są to między innymi: moduły przetworników analogowo-cyfrowych i cyfrowo-analogowych, moduł dysku elektronicznego o wielkiej pojemności (1, 2,...8 MB, 1 MB na pojedynczym pakiecie), moduł sterownika sieci lokalnej (LAN – Local Area Network), a także moduł procesora 16-bitowego. Prowadzone są także prace nad wyprowadzeniem systemu operacyjnego CP/M+ (CP/M wersja 3) oraz rozszerzenia graficznego systemu operacyjnego GSX (Graphics System EXtension).

KRZYSZTOF KAMIŃSKI

[2] Szyna ta, wprowadzona na początku lat siedemdziesiątych przez firmę Hewlett-Packard pod nazwą HP-IB, służy do wymiany informacji i sterowania w systemach pomiarowych. Standard ten został znormalizowany przez IEEE (IEEE Std-488), określony skrótem GPIB oraz zaaprobowany przez IEC w normie IEC-625-1. Również w RWPG powstała norma dotycząca tego systemu (ISP-2). Powyższe normy różnią się przyjętymi rozwiązaniami mechanicznymi.

mikroklan

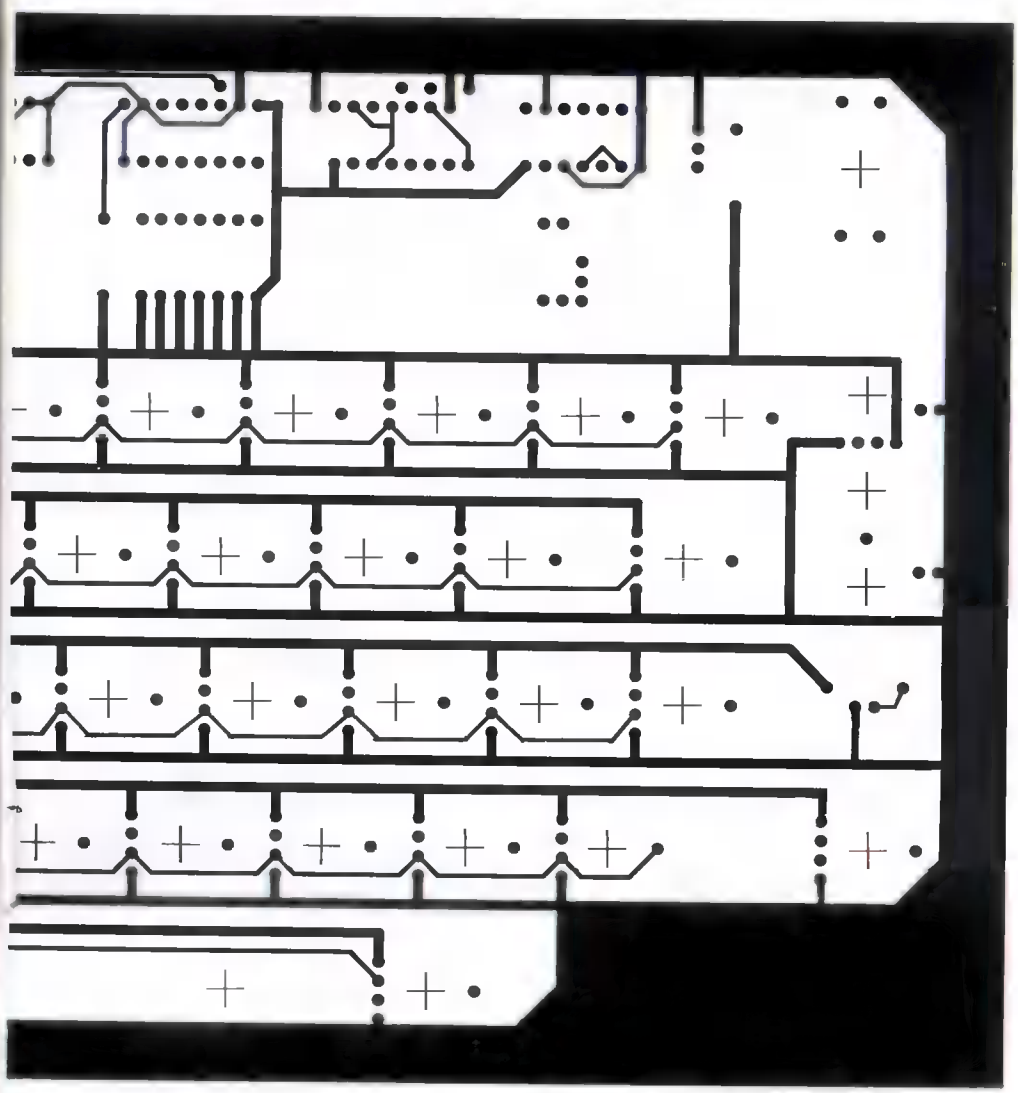
Klawiatura hallotronowa

Wprawdzie są tacy, którzy twierdzą, że „myszka" stanie się kiedyś podstawowym urządzeniem wejściowym komputerów osobistych, ale czasy to jeszcze chyba odległe. Na razie – naciskamy klawisze, stukamy w nie – a czasami nawet, wiedzeni chęcią błyskawicznego „odpalenia" rakiety w kierunku atakującego przybysza z kosmosu, bądź przedpotopowego monstrum – walimy w klawisze z całych sił.

Od klawiatury wymagać będziemy zatem spełnienia kilku warunków: dużej niezawodności mechanicznej, łatwości dołączania do systemu i prawidłowego – z punktu widzenia ergonomii – rozmieszczenia klawiszy. Pierwszy warunek możemy zaspokoić przez prawidłowe podparcie płytki z przyciskami oraz dzięki użyciu trwałych przycisków. Łatwość dołączania oznacza dużą autonomię klawiatury, tzn. taki jej układ elektryczny, który zwalnia jednostkę centralną z obowiązku sterowania podstawowymi funkcjami klawiatury. Ergonomiczność w praktyce polega na podporządkowaniu się jednemu z kilku ogólnie akceptowanych układów klawiatury.

Wydaje się, że przedstawiona poniżej konstrukcja pozwala na spełnienie wszystkich wymagań. Od razu jednak ostrzegamy: nie jest to rozwiązanie najprostsze, budowane z myślą o minimalizacji kosztów i prostocie układowej. Już sam wybór klawiszy hallotronowych praktycznie eliminuje elektroników-hobbistów. Sądzimy jednak, że nawet jeżeli przedstawiona konstrukcja nie doczeka się licznych kopii, to może ona podsunąć kilka ciekawych pomysłów przyszłym konstruktorom klawiatur.

W następnych wydaniach „Mikroklanu" postaramy się przedstawić inne rozwiązania tego fragmentu systemu mikrokomputerowego.



Klawiaturę cechuje duża prostota układu elektronicznego, łatwość reorganizacji układu znaków oraz doskonałe zabezpieczenie przed wygenerowaniem błędnych kodów na skutek drgań zestyków. W klawiaturze zastosowano klawisze z układami scalonymi, których działanie oparte jest na znanym z fizyki zjawisku Halla.

W każdym z klawiszy znajduje się miniaturowy układ scalony SIEMENS SAS 251S4, w którym zawarty jest przerzutnik, reagujący na magnes umieszczony w główce klawisza. Naciskając klawisz zbliżamy magnes do układu scalonego, powodując pojawienie się na wyjściach przerzutnika odpowiednich stanów logicznych. Istnieje kilka wersji przerzutników hallotronowych różniących się wyprowadzeniami. W klawiszach firmy RAFFI, użytych do budowy klawiatury, zastosowano układy o dwóch wyjściach *Q* TTL z otwartym kolektorem. Stan wciśnięcia klawisza sygnalizowany jest pojawieniem się na obu wyjściach stanu logicznego *0*.

Schemat elektroniczny klawiatury przedstawia rysunek. Układ został zbudowany z tanich elementów serii CMOS CD4000 (MOTOROLA MC14000). Większość użytych układów ma swoje odpowiedniki w krajowej serii MCY 74000.

Klawisze ułożone są w matrycy o ośmiu rzędach i siedmiu kolumnach. Wyjścia *Q* przerzutników znajdujących się w klawiszach połączone są w ten sposób, że jedno z nich zwarte jest z odpowiednią kolumną, a drugie z rzędem matrycy. Linie rzędów doprowadzone są do wejść multipleksera *U3*, a linie kolumn do multipleksera *U4*. Oba te układy są typu 8 na 1 (o oznaczeniu MC14512). Wejścia multiplekserów przełą-

czane są z częstotliwością sygnału zegarowego, pochodzącego z generatora zbudowanego z bramek typu *NOR* (U1 – MCY 74001). Wyjście generatora jest podawane poprzez poprawiającą zbocza bramkę z układu *U7* (MCY 74001) na wejścia zegarowe przerzutników typu *D* (U5 – MCY 74013) oraz poprzez bramkę *NOR* (U1) na licznik binarny modulo 256 (U2 – MCY 74520).

W stanie spoczynkowym (żaden z klawiszy nie jest wciśnięty) na wyjściach multiplekserów występują stany wysokie, a zatem do pierwszego przerzutnika wpisywany jest stan niski. Bramka *NOR* (U1 – wyprowadzenia 4, 5, 6) jest otwarta i licznik *U2* adresuje multipleksery. Są one przeglądane kolumnami, tzn. na ich wyjściach pojawiają się kolejno stany klawiszy umieszczonych w pierwszym rzędzie i w pierwszej kolumnie, następnie w pierwszym rzędzie i w drugiej kolumnie itd.

Jednocześnie stan licznika podawany jest na wejścia adresowe pamięci stałej *U6* (2716), która przekształca kod położenia klawisza w matrycy na odpowiedni kod ASCII. W stanie spoczynkowym pamięć jest zablokowana, bowiem na wejściu *CE/* występuje stan wysoki, pochodzący z wyjścia *Q/* pierwszego przerzutnika *D*.

Po wciśnięciu klawisza na odpowiednich liniach rzędów i kolumn matrycy pojawiają się stany niskie. Po wybraniu przez licznik tych właśnie współrzędnych, do pierwszego przerzutnika zostaje wpisany stan wysoki. Ma to następujące konsekwencje:
Po pierwsze – zostaje zablokowany przebieg zegarowy na wejściu licznika, a w rezultacie tego na wejściach adresowych pamięci utrzymywany jest stabilny kod położenia wciśniętego klawisza.
Po drugie – pamięć zostaje uaktywniona niskim poziomem na wejściu *CE/* i na liniach danych pojawia się kod ASCII znaku.
Po trzecie – pierwsze po rozpoznaniu stanu aktywnego klawiatury zbocze sygnału zegarowego wpisuje do drugiego przerzutnika logiczną jedynkę, generując sygnał STROB informujący o gotowości znaku do odczytu. Zwolnienie klawisza powoduje powrót do stanu spoczynkowego.

Zwróćmy uwagę, że częstotliwość zegarowa przerzutników *D* jest osmiokrotnie większa od częstotliwości zmian adresów multiplekserów. Ponadto sygnały na wejściach *CLK* licznika i przerzutników są odwrócone w fazie. Zapewnia to wyeliminowanie zjawiska hazardu przy wpisie do przerzutników.

W klawiaturze, poza zasadniczą matrycą występują klawisze *CTRL* i *SHIFT*. Ich stan jest doprowadzany bezpośrednio do linii adresowych *A6* i *A7* pamięci, wybierając jeden z 64-bajtowych bloków danych. Normalnie linie *A7* i *A6* są w stanie wysokim. Wynika z tego, że „zwykłe" kody ASCII winny być umieszczone w adresach *C0h – FFh*. Wciśnięcie klawisza SHIFT wybiera blok o adresach *40h – 7Fh*, klawisza *CTRL* – blok pomiędzy *80h – BFh*. Blok o adresach *00h – 3Fh*, odpowiadający wciśnięciu obu tych klawiszy nie jest wykorzystywany.

Zawartość pamięci przedstawia wydruk 1. Wymaga on pewnego komentarza. Generowane kody są 7-bitowe i zanegowane; dla przykładu kod spacji *20h* nie występuje w *post. binarnej 00100000* – lecz *x11011111*. Stan linii *b7* odpowiada wybranej przez nas polaryzacji linii *STROB*.

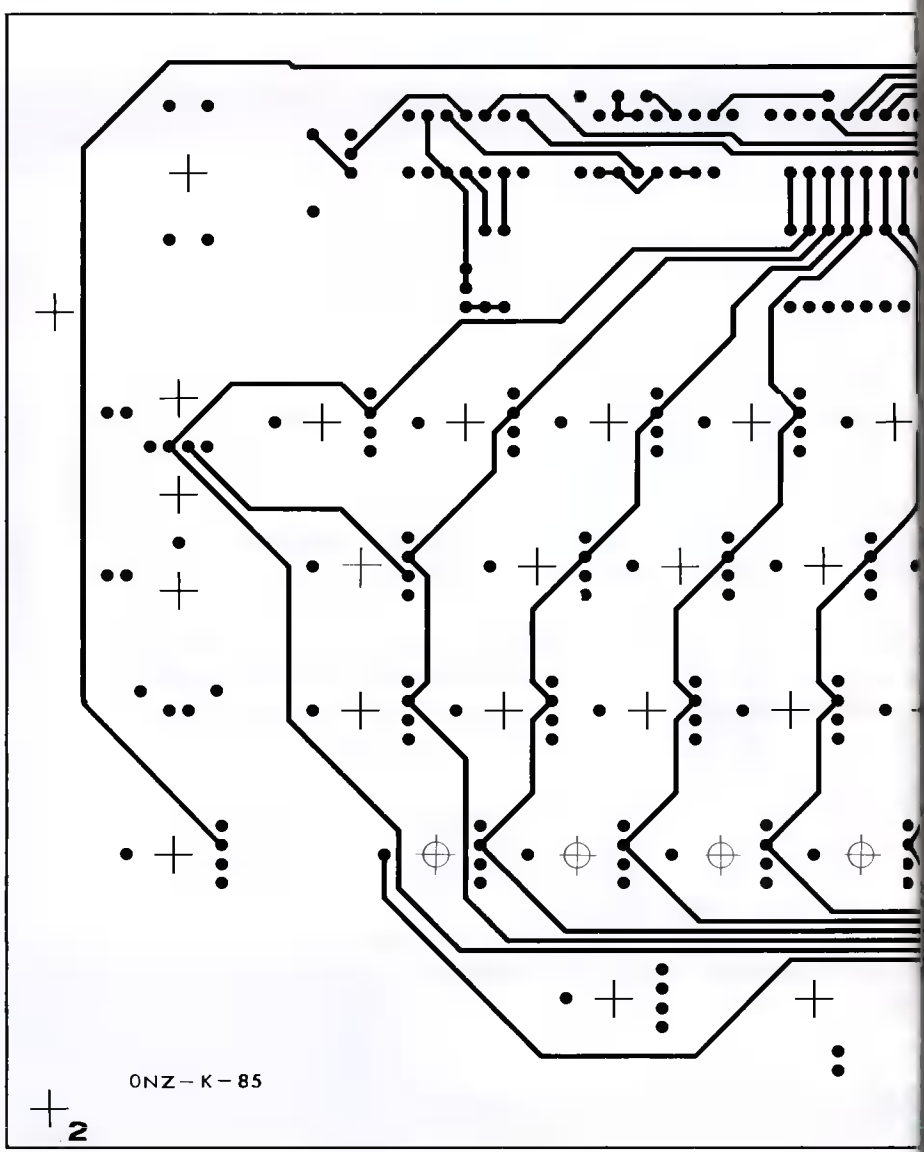
W przedstawionym układzie klawiatury zabrakło miejsca na odrębny klawisz *DEL*. Kod ten (ASCII 7Fh) jest jednak dostępny przy jednoczesnym naciśnięciu klawiszy *SHIFT* i *LF*. Dotyczy to również znaku (kod 5Fh), który generowany jest przez wciśnięcie *CTRL* i *LF*.

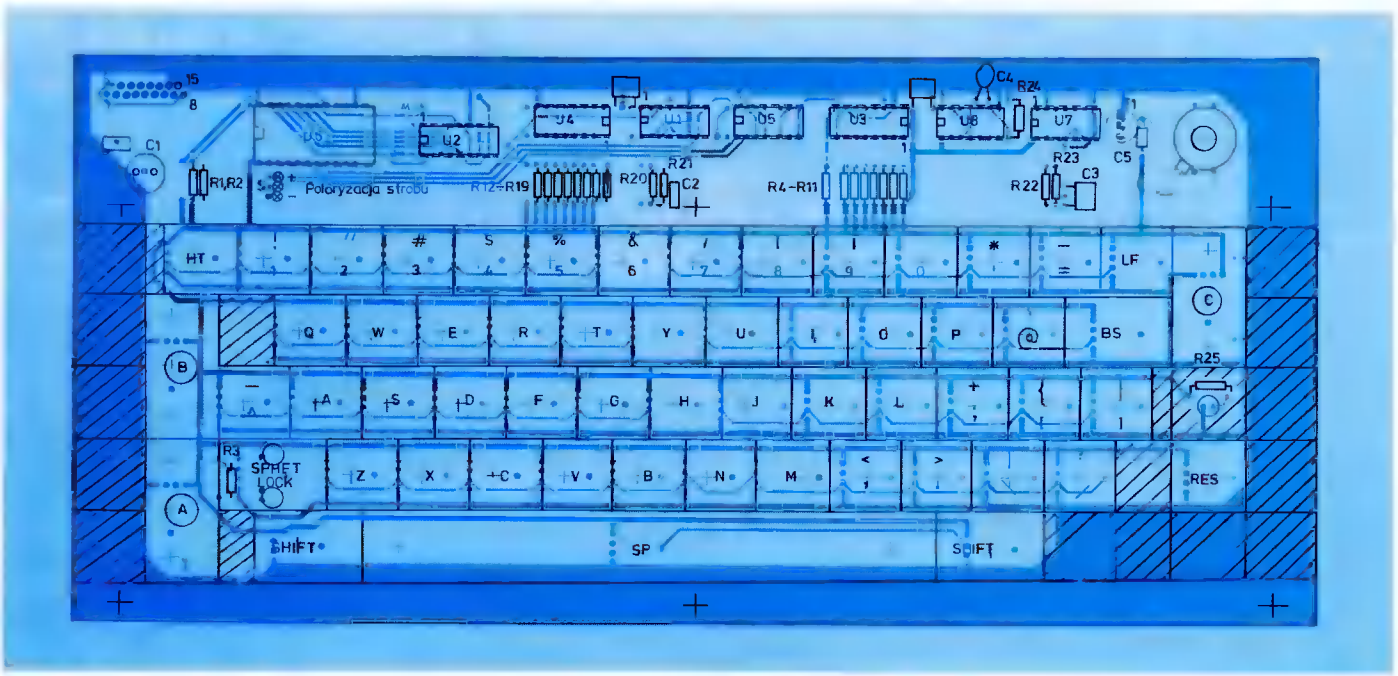
Klawisz *SHIFT* jest zrównoleglony z bistabilnym przyciskiem *LOCK SHIFT*, co pozwala na pisanie małymi literami bez potrzeby ciągłego naciskania dodatkowego klawisza.

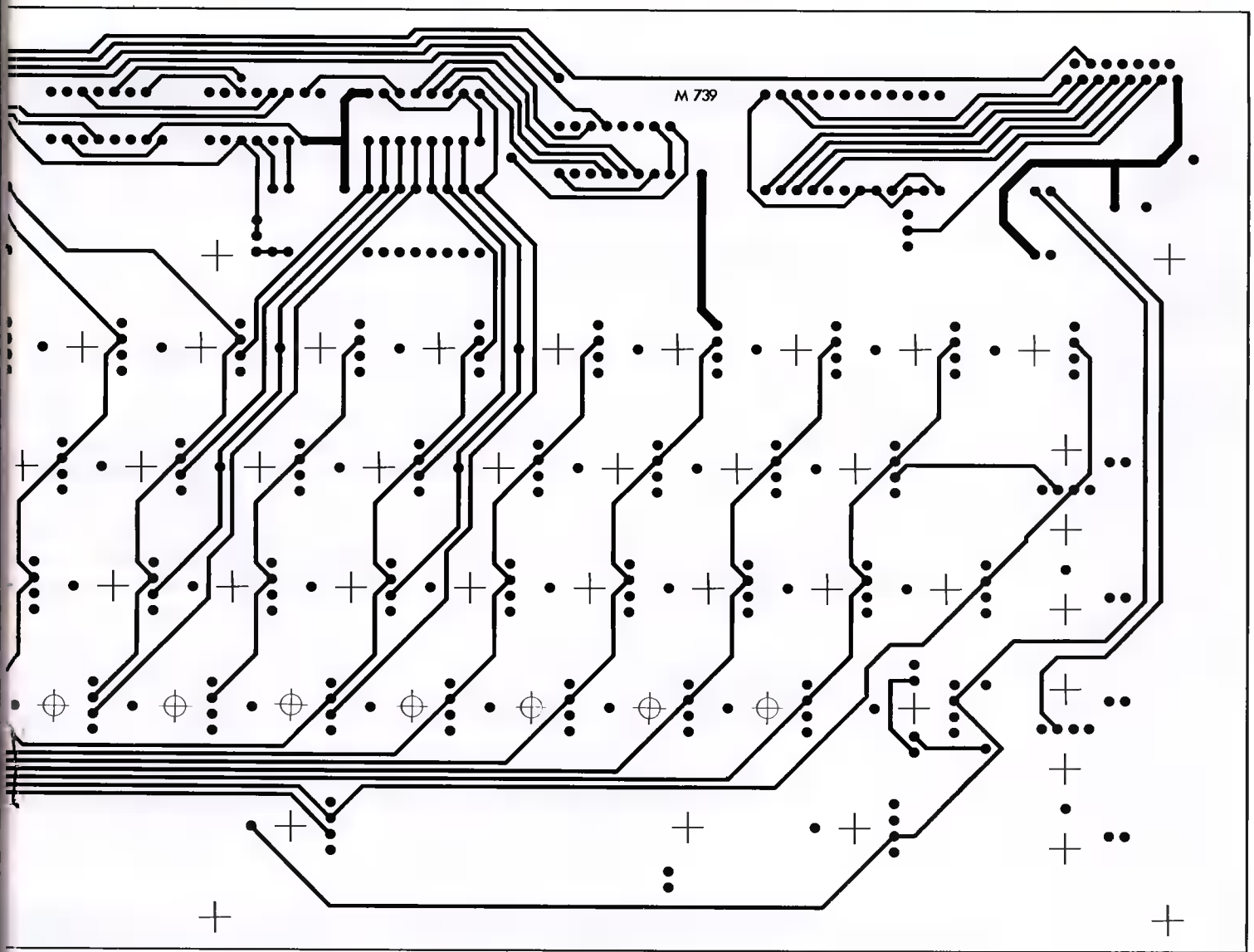
Ponadto zapewniono generowanie sygnału akustycznego po wciśnięciu przycisku. Generator akustyczny o częstotliwości ok. 1 kHz tworzą bramki *NOR* z układu *U7*. Sygnał ten doprowadzany jest poprzez bramkę *NOR* (U7 – wyprowadzenia 4, 5, 6) do miniaturowej słuchawki pracującej jako głośnik. Bramka ta jest otwierana krótkim impulsem, pochodzącym z układu różniczkującego, aktywowanego sygnałem *CE/*.

Klawiatura jest montowana na dwustronnej płytce drukowanej z metalizacją otworów. Na wierzchniej (pierwszej) stronie należy umieścić klawisze wraz z towarzyszącą im „mechaniką" w postaci prowadnic dla przedłużonych klawiszy. Wolne miejsca wypełnione są samoprzylepnymi kostkami z

Rozmieszczenie elementów na płycie klawiatury
Uwagi. Klawisze i diody LED montować na górnej stronie płytki; inne elementy montować na dolnej stronie płytki; powierzchnie zakreskowane nakleić kostki wypełniające




C1
R1,R2
Polaryzacja strobu
R12÷R19
R20
R21
C2
R4-R11
R22
R23
C3
C5
C4
R24
U2
U4
U5
U3
U8
U7
HT
LF
BS
R25
R3
SPRET LOCK
RES
SHIFT
SP
SHIFT


M 739

tworzywa sztucznego, zamykającymi obrys klawiatury. Pod główką klawisza *LOCK SHIFT* montuje się dwie diody LED, sygnalizujące jego położenie. Stan przyłączenia klawiatury do systemu wskazuje świecenie diody *D1*, umieszczonej po prawej stronie konstrukcji.

Układy scalone i elementy bierne należy umieścić po drugiej stronie płytki. Przed przystąpieniem do montażu należy wywiercić otwory pod kołki prowadzące klawisze „(B)" – ESC; „(A)" – CTRL; „(C)" – CR i SP oraz pod wystające bolce pozostałych klawiszy.

Wszystkie układy scalone zasilane są napięciem +5 V pochodzącym z komputera. Pobór prądu nie przekracza kilkudziesięciu (przy niewciśniętym klawiszu) do kilkuset mA.

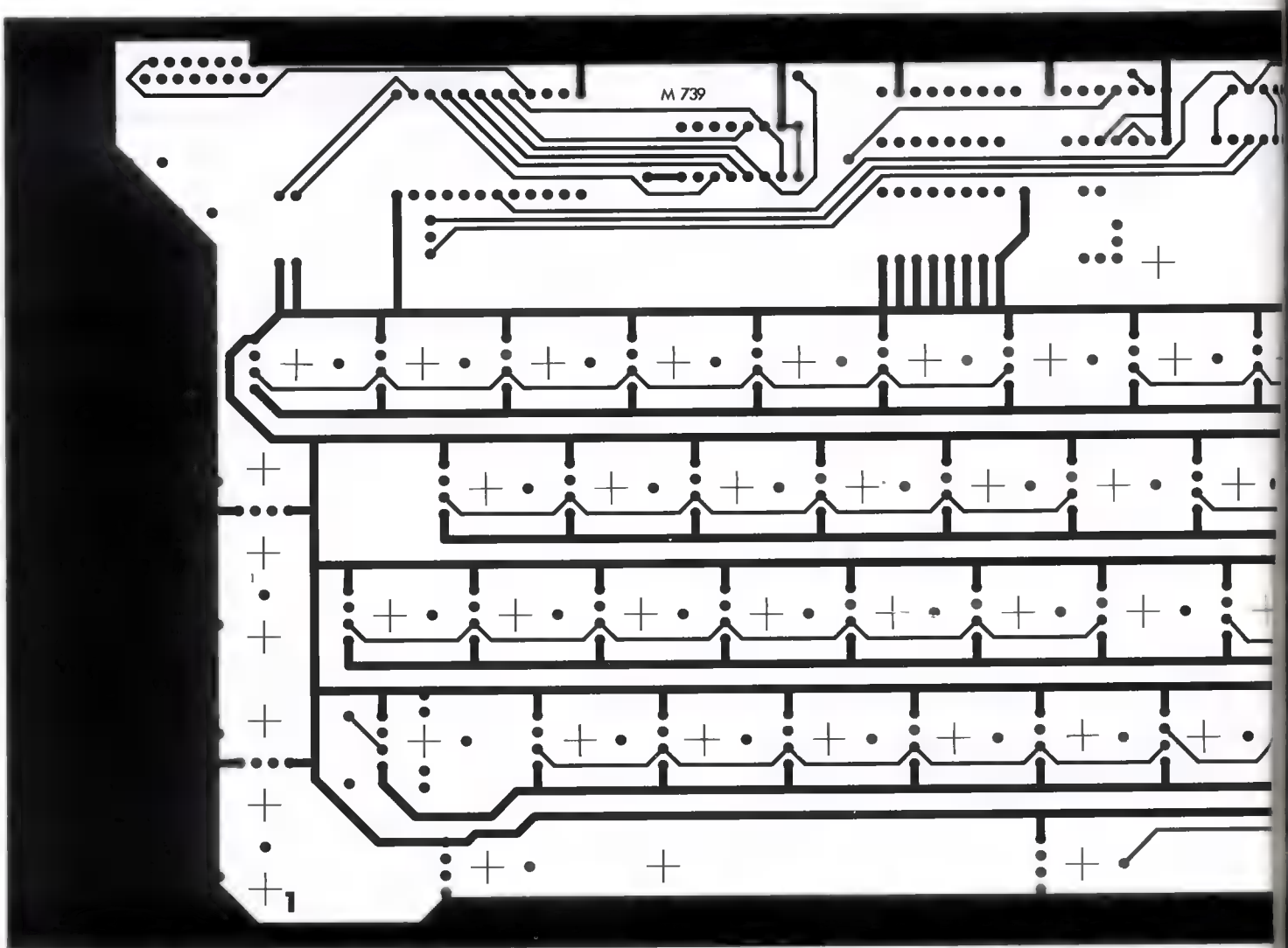
W klawiaturze przewidziano przycisk zerujący komputer – klawisz *ST*. Opornik obciążający jego wyjście winien znajdować się w systemie, na przykład na pakiecie procesora.

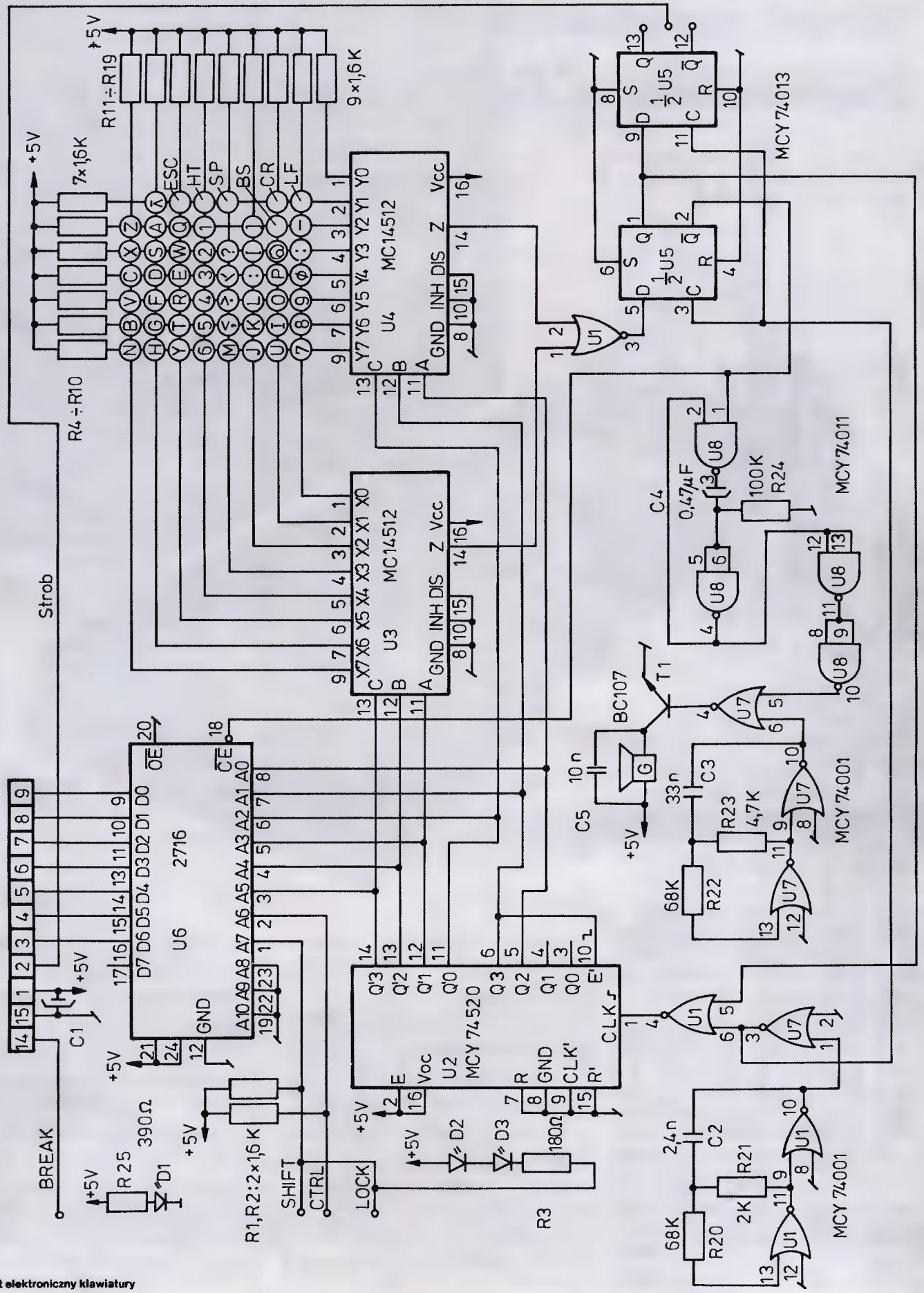
W przyjętym rozwiązaniu stan wciśnięcia klawisza sygnalizowany jest poziomem linii STROB. Polaryzacja tego sygnału zależy od położenia zwory na płytce drukowanej. Aktywny stan tej linii jest utrzymywany tak długo, jak długo wciśnięty jest którykolwiek z klawiszy alfanumerycznych.

Może to być źródłem kłopotów przy pracy z programami zawierającymi procedury badania stanu klawiatury, np. z systemem operacyjnym CP/M. Należy wtedy dodać, poza klawiaturą, układ różniczkujący sygnały STROB.

Wyjście klawiatury nie zawiera buforów. Wynika to z faktu, że w rozwiązaniu autorów znaki z klawiatury wprowadzane są do systemu poprzez port równoległy (i8255, MC6821, Z80PIO), a klawiatura umieszczona jest w bezpośrednim sąsiedztwie portu. W wielu zastosowaniach konieczne będzie dodanie buforów, np. 74 LS 244.

JERZY ORKISZEWSKI
IGNACY NOWOSIELSKI





Schemat elektroniczny klawiatury

Pamięć dynamiczna dla procesora Z80

Rys. 1. Schemat ideowy przykładowego dołączenia pamięci dynamicznych do procesora Z80

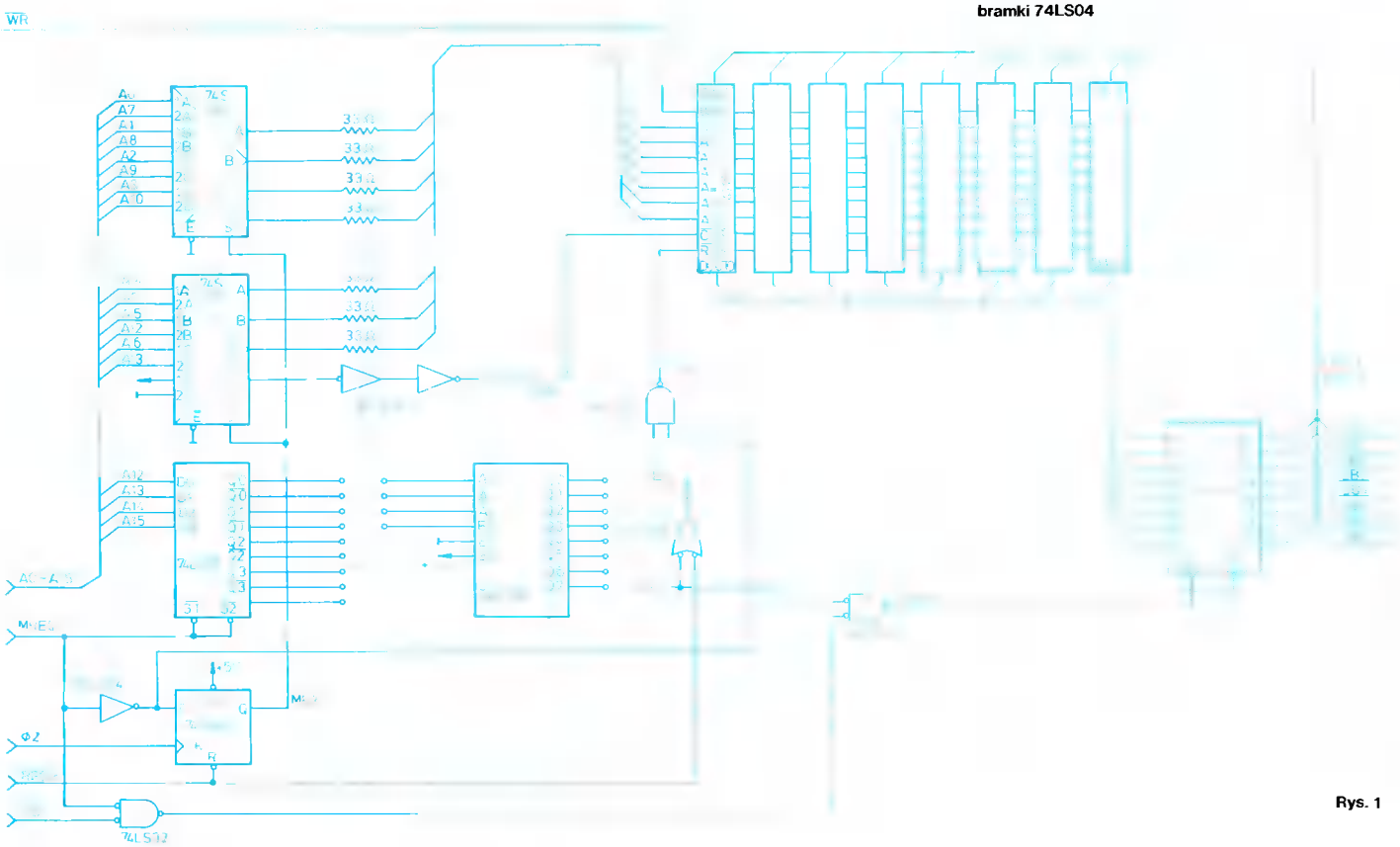
Rys. 2. Krótkotrwały impuls zakłócający w czasie trwania sygnału strobującego RAS

Rys. 3. Zależności czasowe pomiędzy sygnałami 2, MREQ, RAS, MUX, CAS, SZYNĄ DANYCH DB0 – DB7 procesora Z80 podczas operacji odczytu zawartości pamięci

t_1 – czas propagacji przez bramkę 74LS04 i 74LS00

t_2 – czas propagacji przez układ 74S74

t_3 – czas propagacji przez multiplekser 74S157 i dwie bramki 74LS04

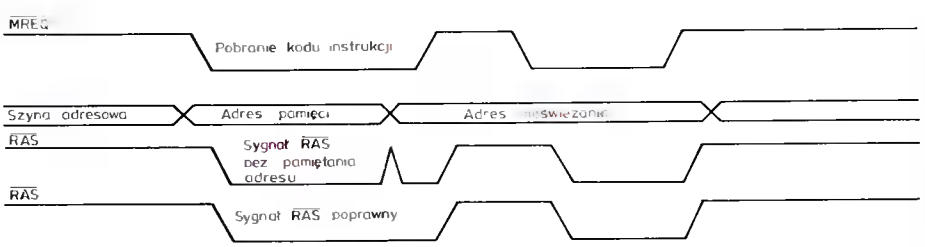


Rys. 1

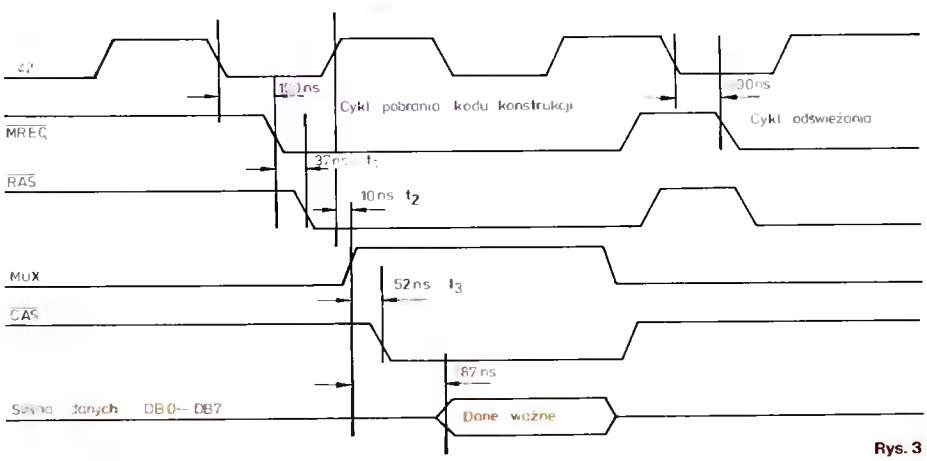
Przyłączenie pamięci dynamicznej do systemu mikroprocesorowego wyposażonego w procesor Z80 jest rzeczą prostą. Wynika to z faktu, że szyna sterująca procesora oraz jego wewnętrzna organizacja logiczna pozwalają na łatwą realizację procesu odświeżania pamięci.

Procesor Z80 został wyposażony w 7-bitowy rejestr odświeżania pamięci, w którym przechowywany jest adres kolejnego wiersza matrycy pamięci. Zawartość tego rejestru jest zmieniana w każdym cyklu pobrania kodu instrukcji.

Cykl pobrania instrukcji FETCH procesora Z80 składa się z dwóch części. W pierwszej procesor pobiera z pamięci programu kod kolejnej operacji do wykonania. W drugiej, analizuje pobraną wartość w celu podjęcia stosownej akcji. Jednocześnie ta druga faza cyklu FETCH wykorzystywana jest w celu odświeżania zawartości pamięci. W tym celu procesor wystawia na szynę adresową adres kolejnego wiersza matrycy pamięci,



Rys. 2



Rys. 3

dokończenie na str. 32

Amstrad CPC 6128

Granice pomiędzy komputerami domowymi (ang. home computers), a komputerami osobistymi (ang. personal computers), zaczynają się coraz bardziej zacierać. Klasycznym przykładem tego jest Amstrad CPC 6128. Czy rzeczywiście ma on już cechy komputera osobistego?

W ciągu zaledwie jednego roku powstała trzecia generacja mikrokomputera CPC. Już „dziadek" CPC 464 był bestsellerem. Jego następca – CPC 664 został wyposażony w stację dyskietek oraz system operacyjny CP/M. Szybko okazało się, że obszar pamięci RAM jest zbyt mały dla większych programów. W CPC 6128 dzięki rozszerzeniu pamięci do 128 KB usunięto ten mankament, a ponadto wyposażono komputer w ulepszony system operacyjny CP/M+.

CPC 6128 składa się z monitora monochromatycznego lub kolorowego i klawiatury. Klawiatura, zaprojektowana wg normy ASCII, jakością nie odbiega od klawiatur komputerów wyższej klasy. Tak istotny dla programu WordStar klawisz *CONTROL* znajduje się wreszcie tam, gdzie być powinien.

W nowym modelu utrzymano bez zmian mikroprocesor Z80 o cyklu 4 MHz. Pojemność pamięci ROM – 48 KB, natomiast RAM – 128 KB, z czego 16 KB służy jako pamięć ekranu.

ZŁĄCZA

Przez interfejs przyłączyc można drugą stację dyskietek. Z uwagi na wyprowadzenie na zewnątrz szyny, zagwarantowano również możliwość dalszej rozbudowy komputera. W interfejsie drukarki nadal jednak brak ósmego bitu, co w wypadku większości typowych drukarek uniemożliwia dostęp do określonych funkcji specjalnych.

Miłosnicy gier komputerowych mogą przekazywać rozkazy również za pomocą drążka sterowniczego (ang. joystick). Istotny jest również interfejs dla magnetofonu kasetowego, jeśli zamierza się przegrać programy CPC 464 na dyskietkę. Wbudowany głośnik zapewnia uzyskanie efektów akustycznych. Siłę głosu można ustawiać na komputerze, a także korzystać z posiadanej instalacji stereofonicznej.

DYSKIETKI

W CPC 6128 zastosowano dyskietki 3-calowe. W odróżnieniu od większych formatów, są one bardzo odporne w użytkowaniu. Sztywna obudowa z tworzywa sztucznego dobrze chroni nośnik, a metalowa zasuwka szczelnie zamyka otwór zapisu-odczytu. Napęd działa precyzyjnie, ok. 20 razy szyb-

dokończenie na str. 32

Pamięć zewnętrzna na minidyskach elastycznych (2)

Poprzedni artykuł z tego cyklu poświęciliśmy napędowi dysków elastycznych 5,25" oraz zasadom dołączenia go do systemu mikrokomputerowego. W tym artykule spróbujemy przedstawić część mikrokomputera, odpowiedzialną za właściwą pracę napędu dysków, którą jest sterownik dysków (ang. disc controller). Zanim jednak przystąpimy do właściwej części artykułu, należałoby przedstawić zasady przechowywania informacji na dysku, czyli sposób fizycznego zapisu danych na dysku.

FORMAT ZAPISU DANYCH

Dane są przechowywane na dysku 5,25" w postaci szeregowych impulsów zapisanych na centrycznych okręgach (ścieżkach). Liczba ścieżek zależy od gęstości zapisu i wynosi standardowo 40 ścieżek przy gęstości 48 ścieżek na cal (TPI – ang. Tracks Per Inch) lub 80 ścieżek przy gęstości 96 TPI. Ścieżki numerowane są kolejno, poczynając od najbardziej zewnętrznej, która z reguły ma numer 0. Informacja na ścieżce przechowywana jest w postaci bloków (sektorów) o długości 256, 512 lub 1024 bajtów. Kolejność oraz numeracja sektorów nie jest istotna, gdyż każdy sektor jest poprzedzony identyfikatorem zawierającym informacje o numerze ścieżki, numerze sektora oraz jego długości. Liczba sektorów na ścieżce zależy od ich długości i przykładowo przy podwójnej gęstości zapisu liczba ta może wynosić 8 lub 9 sektorów 512-bajtowych (po jednej stronie dysku).

Podział całej powierzchni dysku na pola stałe pozwala w łatwy sposób lokalizować informacje na dysku oraz modyfikować dowolny jego fragment, bez niebezpieczeństwa uszkodzenia informacji sąsiednich.

ZADANIA STEROWNIKA

Rolą sterownika dysków w systemie mikroprocesorowym jest zamiana postaci informacji przesyłanych pomiędzy systemem a napędem dysków. Zadanie to tylko z pozoru wygląda prosto. W praktyce liczba funkcji, jakie sterownik musi wykonać dla zrealizowania jednego przesłania, czyni go bardziej skomplikowanym niż niejeden mikroprocesor. Funkcje te zostaną przedstawione na podstawie zapisu sektora, który jest jedną z podstawowych operacji realizowanych przez sterownik.

Od strony procesora transmisja na dysk wygląda następująco:

- przekazanie do sterownika numeru ścieżki i sektora, do którego mają być przesłane dane,
- przesłanie do sterownika (programowo lub za pomocą kanału DMA) odpowiedniej liczby bajtów danych,
- odczytanie (programowo lub po odebraniu przerwania) informacji o ewnetualnych błędach transmisji.

W tym czasie sterownik dysków realizuje następujące czynności:

- przyjmuje od procesora i zapamiętuje w swoich rejestrach numer sektora i ścieżki,
- testuje stan linii *WRITE PROTECT*, sygnalizując błąd w wypadku próby zapisu na dysk z ochroną zapisu,
- przesuwa głowicę napędu dysków na wskazaną ścieżkę, ustawiając linię *DIRECTION*, określającą kierunek przesuwania oraz generująco dpowiednią liczbę impulsów na linii *STEP*,
- po zatrzymaniu głowicy generuje sygnał *HEAD LOAD*, w celu jej opuszczenia na powierzchnię dysku,
- przyjmuje impulsy danych z napędu, zamieniając je na bajty danych, w celu zlokalizowania identyfikatora wskazanego sektora,
- w czasie odczytywania identyfikatora generuje słowo kontrolne, w celu sprawdzenia poprawności odczytu. Wygenerowane słowo porównuje z odczytanym z dysku. W wypadku błędu kontynuuje poszukiwanie sektora. Jeżeli po kilku obrotach dysku (po kilkakrotnym wystąpieniu impulsu na linii *INDEX*) sektor nie zostanie znaleziony, transmisja zostanie przerwana i zasygnalizowany błąd,
- po zlokalizowaniu wskazanego sektora generowany jest sygnał *WRITE GATE*, pobudzający układy zapisu w napędzie dysków,
- programowo lub za pomocą transmisji DMA (sygnały DRQ i DACK) pobierany jest bajt danych z procesora. Bajt ten zamieniany jest na ciąg impulsów (danych i zegara) i przesyłany po linii *WRITE DATA*. Jednocześnie na podstawie bajtu danych generowane jest słowo kontroli cyklicznej,
- po zapisaniu wszystkich bajtów zapisywane jest słowo kontroli cyklicznej,
- generowany jest do procesora sygnał o prawidłowym zakończeniu transmisji.

Operacja zapisu oraz odczytu sektora jest podstawową, a jednocześnie jedną z prostszych czynności realizowanych przez sterowników dysków. Przedstawiona została ona w sposób uproszczony, bez uwzględnienia problemów związanych z separacją impulsów odczytanych z dysku oraz prekompensacją, mającą na celu wstępną korektę zapisywanych impulsów dla zmniejszenia wpływu błędów stałych napędu.

STEROWNIK DYSKÓW WD 2797

Z uwagi na złożoność funkcji realizowanych przez sterownik dysków, jest on konstruowany na podstawie specjalizowanego układu o wielkim stopniu scalenia. Do sterowania napędów dysków 5,25"(pojedynczej lub podwójnej gęstości), najczęściej wykorzystywane są układy serii 179x lub 279x firmy Western Digital lub 8272 firmy Intel (albo jego odpowiednik µPD 765 firmy NEC). Niniejszy artykuł zamierzamy poświęcić sterownikowi dysków zrealizowanemu na układzie WD 2797. Następny artykuł tego cyklu będzie opisywał wykorzystanie układu 8272.

Układ WD 2797 jest kontynuacją serii układów przeznaczonych do sterowania napędami dysków, do której należą 1771 i kolejny 1797. Układ 1771 przeznaczony był do sterowania napędów dysków jednostronnych pojedynczej gęstości i nie jest wykorzystywany w nowo opracowanych konstrukcjach.

Układ 1797 od strony programowej nie różni się od układu 2797, dzięki czemu istniejące procedury obsługi, opracowane dla układu 1797, mogą być wykorzystane do sterowania pracą 2797. Różnica między układami polega na włączeniu do układu 2797 pewnych funkcji sprzętowych, realizowanych poza 1797. Do funkcji tych należą:

- separacja wejściowych impulsów danych za pomocą wewnętrznej pętli fazowej,
- prekompensacja impulsów wyjściowych.

Schemat blokowy układu WD 2797 przedstawiono na rysunku 1. *Rejestr ścieżki* jest przeznaczony do pamiętania numeru ścieżki, nad którą aktualnie znajduje się głowica napędu dysków. Podczas operacji zapisu lub odczytu, zawartość rejestru porównywana jest z odczytanym numerem ścieżki znajdującym się w identyfikatorze sektora. Rejestr może być odczytywany i zapisywany przez procesor.

Rejestr sektora przechowuje numer sektora, którego dotyczy transmisja. Jego zawartość jest również porównywana podczas odczytu identyfikatora sektora. Rejestr może być odczytywany i zapisywany przez procesor. Zawartość rejestrów ścieżki i sektora nie powinny być zmieniane w czasie przesyłania danych między mikrokomputerem a dyskiem elastycznym.

Rejestr statusu zawiera informację o stanie sterownika i napędu dysków oraz o wyniku transmisji. Jego zawartość zależy od typu wykonywanej operacji. Jedynie dwa bity rejestru nie zależą od wykonywanej operacji i oznaczają:

- Bit S0 (najmniej znaczący) określa stan układu 2797. Jeżeli ma on wartość logiczną 0, to nie jest wykonywana zadna operacja, natomiast gdy ma wartość logiczną 1, to operacja jest w trakcie wykonywania i wowczas nic nie należy wysyłać do rejestru rozkazu;
- Bit S7 (najbardziej znaczący) określa gotowość napędu dysków. Jeżeli bit ma wartość logiczną 0, to jednostka dysków jest gotowa do pracy. Jeżeli bit ma wartość logiczną 1, to napęd jest nie wyłączony lub dysk nie został właściwie w nim umieszczony.

Znaczenie pozostałych bitów rejestru statusu przedstawia tabela 3.

Rejestr statusu może być przez procesor jedynie odczytywany.

Rejestr rozkazu zapisywany jest przez procesor i definiuje czynność, jaką ma podjąć sterownik dysków.

Rejestr danych wykorzystywany jest do przekazywania kolejnych bajtów danych podczas operacji odczytu i zapisu oraz do przekazania numeru ścieżki, na ktorą ma przenieść się głowica podczas operacji przesuwania.
Rejestr przesuwny umożliwia tworzenie bajtu danych w oparciu o odczytane z dysku impulsy oraz pozwala zamienić bajt na postać szeregową podczas operacji zapisu.
Układ detekcji znaczników porównuje zawartość odczytanego bajtu danych z wzorcem znacznika, który powinien znajdować się na początku każdego identyfikatora sektora oraz bloku danych.
Układ generacji CRC wytwarza słowo kontroli cyklicznej, porównywane z odczytanym z dysku lub zapisywane za sektorem danych.
Jednostka arytmetyczno-logiczna porównuje numery ścieżki i sektora zapamiętane w rejestrach sterownika z odczytanymi z dysku oraz uaktualnia rejestr ścieżki podczas przesuwania głowicy.
Układ prekompensacji przemieszcza ułożenie impulsów zegara i danych w zależności od ich sekwencji, w celu zmniejszenia błędów stałych jednostki dysków,

Tabela 1. Adresowanie rejestrów układu WD 2797

CS	A1	A0	Odczyt rejestru (RE=0)	Zapis rejestru (WE=0)
1	–	–	nie został wybrany	żaden z rejestrów
0	0	0	statusu	rozkazow
0	0	1	ścieżek	ścieżek
0	1	0	sektorów	sektorów
0	1	1	danych	danych

Tabela 2. Rozkazy układu WD 2797

Typ rozkazu	B7	B6	B5	B4	B3	B2	B1	B0
RESTORE	0	0	0	0	h	V	r1	r0
SEEK	0	0	0	1	h	V	r1	r0
STEP	0	0	1	T	h	V	r1	r0
STEP-IN	0	1	0	T	h	V	r1	r0
STEP-OUT	0	1	1	T	h	V	r1	r0
READ SECTOR	1	0	0	m	1	E	U	0
WRITE SECTOR	1	0	1	m	1	E	U	a0
READ ADDRESS	1	1	0	0	0	E	U	0
READ TRACK	1	1	1	0	0	E	U	0
WRITE TRACK	1	1	1	1	0	E	U	0
FORCE INTER-RUPT	1	1	0	1	I3	I2	I1	I0

Bity r1 i r0 określają czas oczekiwania pomiędzy kolejnymi impulsami na linii STEP:

r1	r0	zegar 1 MHz
0	0	6 ms
0	1	12 ms
1	0	20 ms
1	1	30 ms

Znaczenie pozostałych bitów przedstawiono w części opisowej odpowiedniego rozkazu.

Układ separatora, wykorzystując wewnętrzny układ pętli fazowej, rozdziela odczytany z dysku przebieg, a w konsekwencji odzyskuje impulsy zegarowe i danych oraz eliminuje impulsy fałszywe, wynikające z zakłóceń.

Procesor ma dostęp do sterownika przez 8-bitową szynę danych oraz sygnały sterujące CS, RE, WE, A0 i A1. Za pomocą tych linii może odczytać lub zapisać jeden z rejestrów układu w sposób przedstawiony w tabeli 1.

Układ WD 2797 może współpracować z jednostką dysków 8" lub 5,25", pojedynczej lub podwójnej gęstości oraz z zapisem jednostronnym i dwustronnym. Do wyboru tych opcji służą linie wejściowe 5/8, DDEN oraz bit U (B1) rejestru rozkazów. Możliwa jest dynamiczna zmiana rodzaju współpracującej jednostki dysków. Należy jednak zwrócić uwagę na to, że przy współpracy z napędem 8" do sterownika na wejście CLK powinien być doprowadzony przebieg zegarowy o częstotliwości 2 MHz, a przy współpracy z napędem 5,25" – zegar 1 MHz. Dodatkowo zmiany wymaga układu filtru, sterujący pracą wewnętrznego układu VCO, dołączony do wejścia PUMP.

Sygnały interfejsu z napędem dysków przystosowane są do łączenia kilku jednostek dysków do równoległej magistrali. Układ WD 2797 nie generuje sygnałów *DRIVE SELECT i MOTOR ON*, wybierających jeden z napędów i dlatego sygnały te muszą być generowane przez logikę zewnętrzną. Należy pamiętać, by przełączając napęd, zmienić zawartość rejestru ścieżek, w zależności od tego, gdzie w danym napędzie znajduje się głowica odczytująco-zapisująca.

ROZKAZY UKŁADU WD 2797

Sterownik dysków 2797 akceptuje jedenaście rozkazów. Są one przedstawione w tabeli 2.
Pięć pierwszych rozkazów dotyczy manipulacji głowicą napędu.
Rozkaz RESTORE powoduje przesuwanie głowicy na zewnątrz dysku aż do pojawienia się aktywnego sygnału na linii TRACK 00. Jeżeli po wygenerowaniu 256 kroków nie pojawi się sygnał TRACK 00, to sygnalizowany jest błąd. Jeżeli w kodzie rozkazu **bit h=1,** to nastąpi opuszczenie głowicy. Jeżeli dodatkowo **bit V=1,** to zostanie odczytany numer ścieżki z pierwszego napotkanego identyfikatora sektora.
Rozkaz SEEK umożliwia przesuwanie głowicy na wybraną ścieżkę. Numer ścieżki docelowej jest wpisywany do rejestru danych, a zadaniem układu jest wygenerowanie sygnałów DIRECTION i STEP. Po zakończeniu operacji w rejestrze ścieżek znajduje się numer docelowy. Znaczenie **bitów h** i **V** kodu rozkazu jest takie samo jak w rozkazie poprzednim.
Rozkaz STEP pozwala na przesunięcie głowicy o jedną ścieżkę w takim kierunku, jak przy ostatnio wykonanym rozkazie. Jeżeli **bit T** kodu rozkazu ma wartość 1, to rejestr ścieżek zostanie uaktualniony. Znaczenie **bitów h** i **V** jest takie jak podano powyżej.
Rozkaz STEP-IN realizuje funkcje podobne do rozkazu STEP, z tą jednak różnicą, że głowica jest przesuwana w kierunku środka dysku.
Rozkaz STEP-OUT realizuje funkcję podobną do rozkazu STEP, lecz głowica jest przesuwana zawsze na zewnątrz dysku.

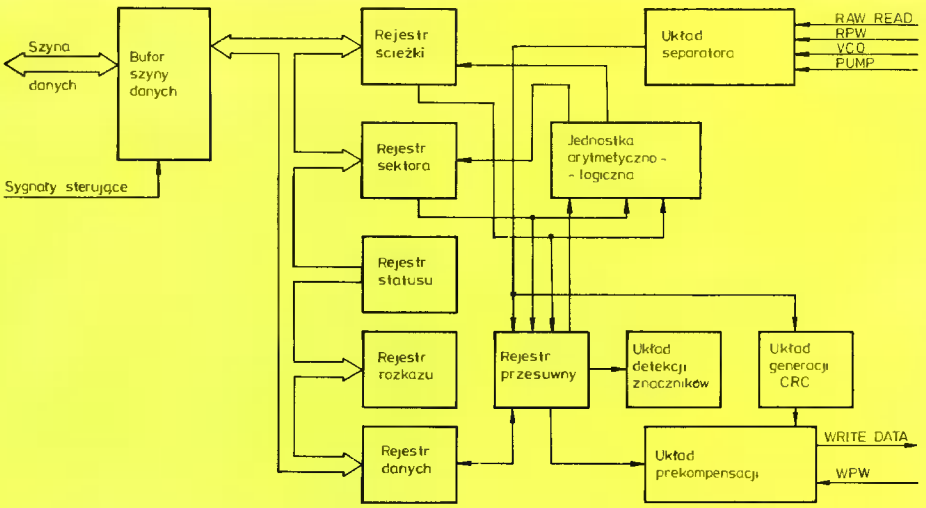
Dwa kolejne rozkazy służą do przesyłania zawartości sektora. Przed ich wykonaniem muszą być zaprogramowane rejestry ścieżek i sektorów.
Rozkaz READ SECTOR przesyła dane ze wskazanego sektora do systemu. Jeżeli sektor nie zostanie znaleziony po 5 obrotach dysku, to sygnalizowany jest błąd. Podczas przesyłania danych jest generowany sygnał DRQ oraz ustawiany **bit S1** rejestru statusu dla każdego bajtu odczytanego. Jeżeli **bit U** kodu rozkazu ma wartość 1, to na linii SIDE ONE SELECT zostanie ustawiony stan wysoki, określający dostęp do drugiej strony dysku. Podczas pierwszej operacji odczytu, po przesunięciu głowicy wskazane jest ustawienie **bitu E=1**, w celu wprowadzenia dodatkowego opóźnienia (30 ms) na uspokojenie głowicy. Jeżeli **bit m=1**, to rozkaz będzie powtarzany dla kolejnego sektora tak długo, aż sektor ten zostanie znaleziony.
Rozkaz WRITE SEKTOR przesyła dane z systemu do wskazanego sektora. Po wysłaniu każdego bajtu generowany jest sygnał DRQ oraz ustawiany **bit S1** rejestru statusu. Znaczenie **bitów U, E** i **m** kodu rozkazu jest analogicznie jak dla rozkazu READ SECTOR.
Rozkaz READ ADDRESS powoduje przesłanie zawartości pierwszego napotkanego identyfikatora sektora do systemu. Przesyłanych jest sześć bajtów, określających kolejno: numer ścieżki, numer strony dysku, nu-

Tabela 3. Znaczenie bitów rejestru statusu

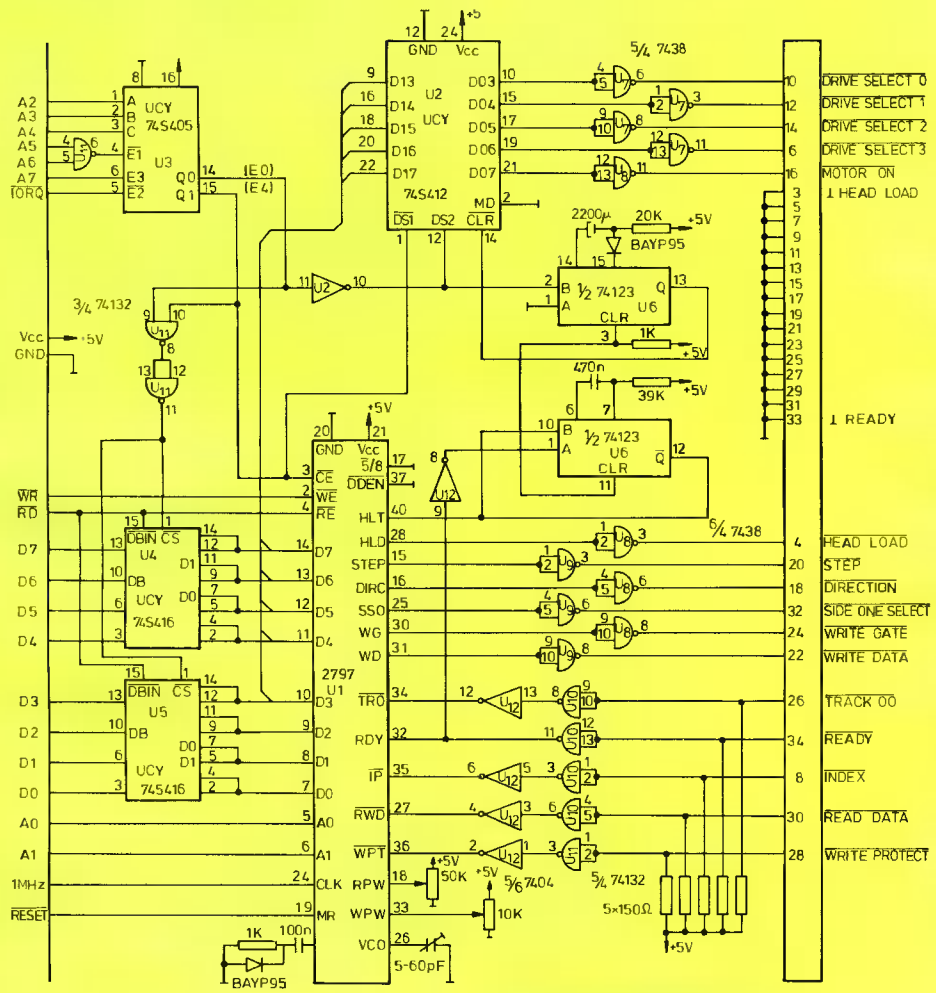
Bit	Przesuw głowicy	READ SECTOR	READ ADDRESS	READ TRACK	WRITE SECTOR	WRITE TRACK
S7	brak gotowości					
S6	ochrona zapisu	0	0	0	ochrona zapisu	
S5	opuszczona głowica	typ rekordu	0	0	0	0
S4	błąd przesuwu	nie ma rekordu		0	nie ma rekordu	0
S3	błąd CRC	błąd CRC	błąd CRC	0	0	0
S2	ścieżka 0	błąd DMA	błąd DMA	błąd DMA	błąd DMA	błąd DMA
S1	INDEX	DRQ	DRQ	DRQ	DRQ	DRQ
S0	zajętość sterownika					

```
FLOPPY 5 1/4" DRIVER   M-80 3.5       03-Jan-86      PAGE  1

                       TITLE FLOPPY 5 1/4" DRIVER
                       ;*********************************
                       ;*     OBSLUGA DYSKOW 5 1/4"     *
                       ;*********************************
                       .Z80
                       ;
                       ; READ/WRITE PHYSICAL - fizyczne operacje
                       ;       na dyskach 5 1/4"
                       ; IN Buffer Disk
                       ; IN Buffer Track
                       ; IN Buffer Sector
                       ; CURrent TRacK
                       ; Disk BUFfer
                       ; DisK ERror Flag
                       PUBLIC  WRPHY,RDPHY
                       EXTRN   IBD,IBT,IBS,CURTRK,DBUFF,DKERF
                       ;
                       ; STALE
                       ;
00E4                   CONTROLLER$Status       EQU     0E4H
00E4                   CONTROLLER$Command      EQU     0E4H
00E5                   CONTROLLER$Track        EQU     0E5H
00E6                   CONTROLLER$Sector       EQU     0E6H
00E7                   CONTROLLER$Data         EQU     0E7H
00E0                   DRIVE$Select$Port       EQU     0E0H
                       ;
0000                   READ$Sign               EQU     000H
0001                   WRITE$Sign              EQU     001H
0088                   FLOPPY$Read$Code        EQU     088H
00A8                   FLOPPY$Write$Code       EQU     0A8H
001C                   FLOPPY$Seek$Code        EQU     01CH
                       ;
0004                   WITH$Waiting            EQU     004H
0080                   MOTOR$On                EQU     080H
0000                   NO$Motor$On             EQU     000H
                       ;
009C                   READ$Errors             EQU     09CH
00DC                   WRITE$Errors            EQU     0DCH
0098                   VERIFY$Errors           EQU     098H
0018                   SEEK$Errors             EQU     018H
                       ;
0000                   BIT$Busy                EQU     0
0001                   BIT$Dma                 EQU     1
0080                   READY$Bit               EQU     080H
0002                   WITH$Waiting$Bit        EQU     2
0001                   DISK$Head$Bit           EQU     1
0001                   DRIVE$Busy              EQU     1
                       ;
0003                   ERROR$Count             EQU     3
                       ;
0000'                  RDPHY:
0000'                  READ$Physical:
0000'  3E 00                   LD      A,READ$Sign
0002'  18 02                   JR      COMMON$Physical
                       ;
0004'                  WRPHY:
0004'                  WRITE$Physical:
0004'  3E 01                   LD      A,WRITE$Sign
                       ;
0006'                  COMMON$Physical:
0006'  32 00FF'                LD      (FLPCOM),A
0009'  3A 0000*                LD      A,(IBD) ; Bit sterujacy dysku
000C'  6F                      LD      L,A
000D'  16 04                   LD      D,WITH$Waiting
000F'  1E 00                   LD      E,NO$Motor$On
0011'  DB E4                   IN      A,(CONTROLLER$Status)
0013'  E6 80                   AND     READY$Bit
0015'  20 0C                   JR      NZ,KNOWN$Drive ; Zaden nie prac
0017'  1E 80                   LD      E,MOTOR$On
0019'  3A 0100'                LD      A,(WRKDR)
001C'  A5                      AND     L       ; Bit wyboru
001D'  28 04                   JR      Z,KNOWN$Drive ; Inny pracuje
001F'  CB 92                   RES     WITH$Waiting$Bit,D
0021'  18 05                   JR      SELECT$Drive    ; Ten sam prac
0023'                  KNOWN$Drive:
0023'  7D                      LD      A,L     ; Drive select
0024'  B3                      OR      E       ; Motor on/off
0025'  32 0100'                LD      (WRKDR),A
0028'                  SELECT$Drive:
0028'  3A 0100'                LD      A,(WRKDR)
002B'  D3 E0                   OUT     (DRIVE$Select$Port),A
002D'  01 7FFF                 LD      BC,07FFFH ; Czekanie na gotow
0030'                  WAIT$For$Ready:
0030'  ED 78                   IN      A,(CONTROLLER$Status)
0032'  E6 80                   AND     READY$Bit
0034'  28 07                   JR      Z,SET$Track
0036'  0B                      DEC     BC      ; Licznik - 1
0037'  78                      LD      A,B
0038'  B1                      OR      C
0039'  20 F5                   JR      NZ,WAIT$For$Ready
003B'  18 3E                   JR      PHYSICAL$Error
003D'                  SET$Track:
003D'  2A 0000*                LD      HL,(CURTRK) ; Aktualna sciezka
0040'  7E                      LD      A,(HL)
0041'  D3 E5                   OUT     (CONTROLLER$Track),A
0043'  3A 0000*                LD      A,(IBT)
0046'  77                      LD      (HL),A  ; Nowa aktualna sciezka
0047'  3A 0000*                LD      A,(IBS)
004A'  FE 0A                   CP      10
004C'  38 04                   JR      C,SET$Sector
004E'  D6 09                   SUB     9
0050'  CB CA                   SET     DISK$Head$Bit,D
0052'                  SET$Sector:
0052'  D3 E6                   OUT     (CONTROLLER$Sector),A
0054'  1E 03                   LD      E,ERROR$Count
0056'                  PHYSICAL$Operation:
0056'  3A 0100'                LD      A,(WRKDR)
0059'  D3 E0                   OUT     (DRIVE$Select$Port),A

005B'  DB E5                   IN      A,(CONTROLLER$Track)
005D'  47                      LD      B,A
005E'  3A 0000*                LD      A,(IBT)
0061'  B8                      CP      B
0062'  C4 0090'                CALL    NZ,SEEK$Command
0065'  20 0E                   JR      NZ,PHYSICAL$Recalibrate
0067'  21 0000*                LD      HL,DBUFF
006A'  CD 00B1'                CALL    PHYSICAL$Command
006D'  C8                      RET     Z       ; Bez bledu
006E'  CB 92                   RES     WITH$Waiting$Bit,D
0070'  3E 02                   LD      A,ERROR$Count-1
```

```
FLOPPY 5 1/4" DRIVER   M-80 3.5       03-Jan-86      PAGE  2

0072'  BB                      CP      E
0073'  20 03                   JR      NZ,PHYSICAL$Error$Test
0075'                  PHYSICAL$Recalibrate:
0075'  CD 00BA'                CALL    RECAL$Command
0078'                  PHYSICAL$Error$Test:
0078'  1D                      DEC     E
0079'  20 05                   JR      NZ,PHYSICAL$Operation
007B'                  PHYSICAL$Error:
007B'  3E 01                   LD      A,1
007D'  32 0000*                LD      (DKERF),A
0080'  C9                      RET             ; Powrot z bledem
0081'                  ;
0081'  3A 00FF'        PHYSICAL$Command:
0084'  FE 01                   LD      A,(FLPCOM)
0086'  28 41                   CP      WRITE$Sign
0088'  18 1B                   JR      Z,WRITE$Command
                               JR      READ$Command
                       ;
                       ; RECAL - Przesun na siez. 0
                       ; SEEK - Przesun na sciezke
                       ;       Drive - wlaczany przed wywolan
                       ;       Rej.A - numer sciezki
                       ;
008A'                  RECAL$Command:
008A'  3E FF                   LD      A,0FFH
008C'  D3 E5                   OUT     (CONTROLLER$Track),A
008E'  3E 00                   LD      A,0
0090'                  SEEK$Command:
0090'  D3 E7                   OUT     (CONTROLLER$Data),A
0092'  3E 1C                   LD      A,FLOPPY$Seek$Code
0094'  D3 E4                   OUT     (CONTROLLER$Command),A
0096'  06 0A                   LD      B,10    ; Czekaj busy
0098'  10 FE                   DJNZ    $
009A'                  WAIT$For$Seek:
009A'  DB E4                   IN      A,(CONTROLLER$Status)
009C'  CB 47                   BIT     BIT$Busy,A
009E'  20 FA                   JR      NZ,WAIT$For$Seek
00A0'  CB D2                   SET     WITH$Waiting$Bit,D
00A2'  E6 18                   AND     SEEK$Errors
00A4'  C9                      RET
                       ;
                       ;
                       ; READ - Czytaj do bufora
                       ;       Drive - wlaczony, ustawiona sciez
                       ;       i sektor
                       ;       Rej.D - strona dysku
                       ;       Rej.HL- adres bufora
                       ;
00A5'                  READ$Command:
00A5'  3E 88                   LD      A,FLOPPY$Read$Code
00A7'  B2                      OR      D
00A8'  D3 E4                   OUT     (CONTROLLER$Command),A
00AA'  06 0A                   LD      B,10
00AC'  10 FE                   DJNZ    $       ; Czekaj busy
00AE'  0E E7                   LD      C,CONTROLLER$Data
00B0'  F3                      DI
00B1'                  NEXT$Read:
00B1'  DB E4                   IN      A,(CONTROLLER$Status)
00B3'  FE 01                   CP      DRIVE$Busy
00B5'  28 FA                   JR      Z,NEXT$Read
00B7'  ED 40                   IN      B,(C)   ; Chyba dana
00B9'  CB 4F                   BIT     BIT$Dma,A
00BB'  28 04                   JR      Z,READ$No$Busy
00BD'  70                      LD      (HL),B
00BE'  23                      INC     HL
00BF'  18 F0                   JR      NEXT$Read
00C1'                  READ$No$Busy:
00C1'  CB 47                   BIT     BIT$Busy,A
00C3'  20 EC                   JR      NZ,NEXT$Read
00C5'  FB                      EI
00C6'  E6 9C                   AND     READ$Errors
00C8'  C9                      RET             ; Z=0 - Bez bledu
                       ;
                       ; WRITE - zapis z bufora
                       ;       Drive - wlaczony, ustawiona sciez
                       ;       i sektor
                       ;       Rej.D - strona dysku
                       ;       Rej.HL- adres bufora
                       ;
00C9'                  WRITE$Command:
00C9'  3E A8                   LD      A,FLOPPY$Write$Code
00CB'  B2                      OR      D
00CC'  D3 E4                   OUT     (CONTROLLER$Command),A
00CE'  06 0A                   LD      B,10
00D0'  10 FE                   DJNZ    $       ; Czekaj na busy
00D2'  0E E7                   LD      C,CONTROLLER$Data
00D4'  F3                      DI
00D5'                  NEXT$Write$Data:
00D5'  46                      LD      B,(HL)
00D6'  23                      INC     HL
00D7'                  NEXT$Write:
00D7'  DB E4                   IN      A,(CONTROLLER$Status)
00D9'  FE 01                   CP      DRIVE$Busy
00DB'  28 FA                   JR      Z,NEXT$Write
00DD'  ED 41                   OUT     (C),B
00DF'  CB 4F                   BIT     BIT$Dma,A
00E1'  20 F2                   JR      NZ,NEXT$Write$Data
00E3'  CB 47                   BIT     BIT$Busy,A
00E5'  20 F0                   JR      NZ,NEXT$Write
00E7'  FB                      EI
00E8'  E6 DC                   AND     WRITE$Errors
00EA'  C0                      RET     NZ
00EB'  3E 81                   LD      A,FLOPPY$Read$Code
00ED'  B2                      OR      D       ; strona dysku
00EE'  D3 E4                   OUT     (CONTROLLER$Command),A
00F0'  06 0A                   LD      B,10    ; Czekaj na busy
00F2'  10 FE                   DJNZ    $
00F4'                  VERIFY$Sector:
00F4'  DB E7                   IN      A,(CONTROLLER$Data)
00F6'  DB E4                   IN      A,(CONTROLLER$Status)
00F8'  CB 47                   BIT     BIT$Busy,A
00FA'  20 FB                   JR      NZ,VERIFY$Sector
00FC'  E6 98                   AND     VERIFY$Errors
00FE'  C9                      RET             ; Z=0 - bez bledu
                       ;
00FF'  00              FLPCOM: DB      0
0100'  00              WRKDR:  DB      0
                       ;
                               END

No Fatal error(s)
```



Rys. 1. Schemat blokowy układu WD 2797



Rys. 2. Schemat ideowy sterownika dysków 5,25"

mer sektora, długość sektora, 2 bajty słowa kontrolnego. Znaczenie **bitów U** i **E** kodu rozkazu jest takie jak w poprzednich rozkazach.

Rozkaz READ TRACK powoduje przesłanie do systemu całej zawartości ścieżki, obejmującej wszystkie znaczniki, identyfikatory, przerwy międzyblokowe oraz dane. Przesłanie to rozpoczyna się po pojawieniu się sygnału INDEX. Podczas przesyłania informacji nie jest dokonywana żadna kontrola. Ciąg danych jest synchronizowany w chwili wykrycia znaczników. Znaczenie **bitów U** i **E** kodu rozkazu jest takie samo, jak w poprzednich rozkazach.

Rozkaz WRITE TRACK pozwala na fizyczne formatowanie ścieżki na dysku. Przesłane dane zapisywane są kolejno: od napotkania impulsu na linii INDEX do kolejnego pojawienia się tego znacznika. Zawartość całej ścieżki, zawierająca wszystkie znaczniki, identyfikatory, przerwy międzyblokowe oraz bloki danych, powinna znajdować się w pamięci systemu. Taka postać rozkazu pozwala formatować dyski w dowolnym standardzie (np. IBM 3740, SYSTEM 34). **Bity U** i **E** kodu rozkazu pełnią taką samą funkcję jak w poprzednich rozkazach.

Rozkaz FORCE INTERRUPT pełni dwie funkcje. przerywa aktualnie wykonywany rozkaz (w szczególności jeżeli jest rozkaz przesłania wielu sektorów) oraz ustawia warunki, jakie muszą być spełnione, by sterownik generował przerwanie. Znaczenie kolejnych bitów kodu rozkazu jest następujące:

10 – generuj przerwanie, gdy pojawi się sygnał gotowości,

11 – generuj przerwanie, gdy zniknie sygnał gotowości,

12 – generuj przerwanie po każdym sygnale INDEX,

13 – generuj przerwanie po zakończeniu każdego rozkazu.

Tabela podaje znaczenie bitów rejestru statusu po wykonaniu wyżej opisanych rozkazów.

UKŁAD STEROWNIKA DYSKÓW 5,25"

Proponowany układ sterownika dysków (rys. 2) został zrealizowany z wykorzystaniem sterownika WD 2797. Współpracuje on z mikroprocesorem Z80 (2,5 MHz). Dla zmniejszenia kosztów zrezygnowano z kanału DMA i obsługa sterownika, łącznie z przekazywaniem i odbieraniem danych, jest dokonywana programowo. Do sterownika można dołączyć 4 jednostki dwustronnych, podwójnej gęstości (DSDD) dysków 5,25". Sygnały wyboru napędu są wytwarzane przez program obsługi za pośrednictwem układu 8212 (UCY 74S12). Układ 8212 służy również do wytwarzania sygnału MOTOR ON, którego zadaniem jest podtrzymanie obrotów uprzednio wybranego dysku w wypadku przełączenia na inny dysk (np. podczas kopiowania z dysku na dysk). Rozwiązanie to spełnia wymagania jedynie w wypadku istnienia dwóch napędów dyskow. W innym wypadku niepotrzebnie włączane są obroty kolejnych dysków i należałoby z tego sygnału zrezygnować. Rejestr wyboru napędu zerowany jest przez dodatkowy uniwibrator po upływie około 20 sekund od ostatniej operacji na dyskach. Ma to na celu z jednej strony wymuszenie obrotów dysku (i dołączenie sygnałów interfejsu) w czasie krótkich przerw pomiędzy kolejnymi operacjami na dyskach, a z drugiej wyłączenie dysków, jeżeli system przez dłuższy czas do nich się nie odwołuje.

Druga połowa układu 74123 jest wykorzystana do wytwarzania opóźnionego (o około 100 ms) sygnału HLT, oznaczającego opusz-

czenie głowicy w odpowiedzi na sygnał HEAD LOAD, pod warunkiem, że jednostka dysków sygnalizuje gotowość. Dalsza część układu sprowadza się do buforowania bramkami mocy typu *„otwarty kolektor"* sygnałów wyjściowych sterownika oraz buforowania bramkami z wejściem Schmitta sygnałów wejściowych.

Na omawianym schemacie przedstawiono dekoder adresów lokalizujący układy w przestrzeni adresowej wejścia-wyjścia oraz bufory systemowej szyny danych, pozwalające dołączyć sterownik do rozbudowanego systemu mikroprocesorowego.

PROGRAM OBSŁUGI

Przedstawiony wydruk obejmuje podstawową część programu obsługi, w której skład wchodzą procedury fizycznego odczytu i zapisu danych na wskazany dysk, wybraną ścieżkę i sektor. Przed wywołaniem procedury *READ$Physical* lub *WRITE$Physical*, program obsługi powinien wpisać do zmiennej *IBD* maskę wyboru dysku (08H dla dysku 0, 10H dla dysku 1 itd.), do zmiennej *IBT* numer ścieżki oraz do zmiennej *IBS* numer sektora, który ma zostać przesłany, natomiast do zmiennej *CURTRK* numer ścieżki, nad którym aktualnie znajduje się głowica wybranego dysku. Powinna również zostać wyzerowana zmienna *DKERF*, przez którą przekazywany jest sygnał błędu operacji dyskowej.

Procedury obsługi mają lokalną zmienną, do której zapisywana jest maska wybranego dysku *WRKDR*. Na początku następuje sprawdzenie, czy jeden z napędów jest włączony. Jeżeli gotowy jest inny dysk, to oprócz wybranego sygnału *DRIVE SELECT* generowany jest sygnał *MOTOR ON*. Jeżeli żaden dysk nie pracuje, to program oczekuje na pojawienie się sygnału gotowości. Po włączeniu dysku następuje zaprogramowanie rejestrów ścieżki i sektora. Sektory na ścieżce są numerowane od 1 do 9. Jeżeli numer sektora przekracza 9, interpretowane jest to jako dostęp do sektora od 1 do 9 po drugiej stronie dysku.

Po zaprogramowaniu rejestrów następuje przesunięcie głowicy na właściwą ścieżkę oraz próba transmisji sektora do lub z bufora *DBUFF*. Jeżeli wystąpił błąd, to rozkaz jest powtarzany trzykrotnie (należy zaznaczyć, że sterownik automatycznie powtarza nieudane próby 5 razy), przy czym za drugim razem rozkaz jest poprzedzony przesunięciem głowicy na ścieżkę 0, w celu wyeliminowania błędu niewłaściwego ustawienia głowicy. Jeżeli żadna z prób nie zakończy się pozytywnie, to zmienna *DKERF* przyjmuje wartość 01H.

ADAM PLUTA

System dyskowy MUEL-85

Mniej więcej przed dwoma laty mikrokomputer ZX Spectrum stał się szlagierem polskiego rynku. Wkrótce zaczęto go używać także do celów profesjonalnych (edycja tekstów, sterowanie urządzeń pomiarowych, obliczenia numeryczne itp.). W porównaniu z innymi mikrokomputerami ZX Spectrum dawał niezłe możliwości graficzne, szybki i niezawodny sposób korzystania z magnetofonu, dźwięk, olbrzymią ilość oprogramowania i wyróżniał się stosunkowo niską ceną. Problemy zaczęły się pojawiać dopiero wtedy, gdy nie wystarczał już magnetofon i drukarka ZX Printer.

Clive Sinclair od początku kampanii reklamowej zapowiadał, że wkrótce będzie dostępna szybka pamięć masowa do ZX Spectrum. Większość osób sądziła, że chodzi o 3-calowe dyski. Kiedy jednak po ponad rocznym oczekiwaniu pojawiły się niedoskonałe pamięci Microdrive (jak zapewne wszyscy już wiedzą kaseta Microdrive zawiera zamkniętą pętlę taśmy magnetycznej o jakości zbliżonej do taśmy wideo), niezbędne stało się opracowanie dla ZX Spectrum systemu dyskowego. Kilka takich systemów powstało w Wielkiej Brytanii, ale wszystkie miały wady: przede wszystkim obciążały pamięć operacyjną komputera oraz były niewygodne w użyciu.

W ten sposób doszło do opracowania przez zakład rzemieślniczy MUEL systemu dyskowego.

Interfejs MUEL-85 mieści się w czarnej, płaskiej obudowie o kształcie bardzo przypominającym mikrokomputer ZX Spectrum. Skonstruowany jest tak, aby mieścił się dokładnie pod mikrokomputerem, z którym jest powiązany poprzez złącze krawędziowe. Na płycie czołowej znajdują się dwa przyciski: *RESET*, który wywołuje efekt identyczny do wyłączenia komputera z sieci oraz *NMI*, powodujący wysłanie do mikroprocesora Z80 sygnału przerwania niemaskowalnego.

Na płycie tylnej znajduje się:

- wyjście na drukarkę dowolnego typu; dostępne są wszystkie wyjścia układu 8255 oraz napięcia 5V i 12V (wybór rodzaju transmisji dokonuje się odpowiednią instrukcją rozszerzonego języka BASIC)
- wyjście na cztery napędy dyskowe
- wyjście na monitor monochromatyczny
- przedłużenie złącza mikrokomputera ZX Spectrum
- wejście zasilające, przez które jest zasilany zarówno komputer, jak i interfejs.

Mechanizm napędów dysków jest zasilany za pomocą oddzielnego kabla doprowadzonego z zasilacza.

Rozszerzenie języka BASIC jest zrealizowane za pomocą mechanizmu dynamicznie przełączanych pamięci ROM ZX Spectrum i systemu dyskowego. Całe wewnętrzne oprogramowanie interfejsu jest zawarte w dwóch pamięciach ROM o pojemności 2 oraz 8 KB. Do sterowania napędów dysków zastosowano jednoukładowy sterownik FD 1791. Zmienne systemowe oraz 512-bajtowy bufor znajdują się w pamięci RAM o pojemności 1 KB. Pamięć ta jest umiejscowiona w przestrzeni adresowej między adresami 14 592 i 15 616, wykorzystując w ten sposób nieużywany fragment pamięci RAM mikrokomputera.

Pamięć ROM o pojemności 10 KB zawiera następujące moduły programowe:

- rozszerzenie języka BASIC związane z obsługą dysku
- rozszerzenie język BASIC związane z rozbudową systemu strumieni i kanałów ZX Spectrum
- rozszerzenie języka BASIC związane z dodaniem nowych rozkazów edycyjnych
- oprogramowanie systemu haseł
- oprogramowanie obsługi przycisku NMI (przerwanie niemaskowalne)
- inne rozszerzenia

ROZSZERZENIA JĘZYKA BASIC ZWIĄZANE Z OBSŁUGĄ DYSKU

Nowe instrukcje systemu dyskowego MUEL-85 są kompatybilne z istniejącymi instrukcjami ZX Spectrum, przy czym gwiazdka po słowie kluczowym zapewnia odróżnienie ich od rozkazów taśmowych. Dla użytkownika dostępna jest cała pamięć Spectrum. Jeśli zamierza się skorzystać z programów typu Mega-Basic, które wykorzystują podobny sposób rozszerzenia języka BASIC do zastosowanego w intrfejsie, to za pomocą jednej instrukcji można odłączyć wszystkie rozszerzenia systemu, jakie zapewnia MUEL-85.

Oprócz dublowania wszystkich starych rozkazów taśmowych, wprowadzony został system plików sekwencyjnych. Każdy oddzielny, otwarty na plik sekwencyjny, strumień wymaga około 500 bajtów pamięci Spectrum. Należy jednak pamiętać, że w trybie taśmowym interfejs nie zajmuje pamięci. Oznacza to, że w zasadzie dowolny program działający na zwykłym ZX Spectrum może być wczytywany z dysku. Przycisk *NMI*

znacznie ułatwia wykonanie operacji związanych z przenoszeniem oprogramowania z taśmy na dysk.

Szybkość transmisji wynosi 125 kb na sekundę, jednak niezbędne podziały informacji na sektory i ścieżki powodują, że efektywna szybkość wynosi 32 kb na sekundę. Innymi słowy wczytanie bloku programu lub danych o długości 32 KB trwa ok. 8 s.

Czas zapisywania jest dłuższy, ponieważ zapisane sektory są kontrolowane za pomocą systemu **CRC** (Cyclic Redundancy Check). W wypadku wykrycia niezgodności **CRC**, system dyskowy skreśla z mapy bitowej sektory uszkodzone oraz powtarza nagranie. Oprócz tego w trakcie formatowania dyskietki system wychwytuje i skreśla te sektory, które zostały odczytane z błędem. Taki podwójny system kontroli zapewnia bardzo wysoki stopień bezpieczeństwa zapisu nawet na dyskietkach gorszej jakości.

System dyskowy MUEL-85 został pomyślany tak, aby działał w zasadzie z każdym dyskiem 5-calowym lub mniejszym o standardzie złącza zbliżonym do złącza Shugart. Istnieje możliwość dopasowania interfejsu do posiadanych dysków za pomocą instrukcji określających częstość kroków silnika krokowego oraz liczbę ścieżek na dysku w zakresie od 3 do 255.

System pracuje z pojedynczą gęstością zapisu w standardzie IBM 3740, czyli FM (Frequency Modulated). Stosując napędy 2×80 ścieżek, których cena w Europie Zachodniej wynosi obecnie ok. 150 dol. uzyskuje się dostęp do ok. 400 KB pamięci użytkowej. Stosując dwa takie napędy (lub cztery napędy jednostronne) uzyskuje się więc pojemność ok. 800 KB.

Dostęp do dowolnego zbioru jest prawie natychmiastowy. Dla dysków z czasem kroku 6 ms, czas potrzebny na przejście wszystkich 80 ścieżek wynosi 0,48 s. Odszukanie informacji w skorowidzu jest równie szybkie.

ROZSZERZENIE JĘZYKA BASIC ZWIĄZANE Z ROZBUDOWĄ SYSTEMU STRUMIENI I KANAŁÓW ZX SPECTRUM

W oryginalnym systemie operacyjnym ZX Spectrum wprowadzono namiastkę systemu strumieniowo-kanałowego. Niestety koncepcja ta nie została w pełni zrealizowana. Jedyną użyteczną funkcją jest przełączanie strumienia ekranu na kanał drukarki i odwrotnie.

W interfejsie MUEL wprowadzono następujące kanały:
F – plik sekwencyjny na dysku
M – plik sekwencyjny w pamięci Spectrum
P – drukarka poprzez jej wbudowany interfejs (oparty na układzie 8255).

Zapisu i odczytu z plików dokonuje się rozkazami *INPUT#*, *INKEY#*, *PRINT#* oraz takimi, które wyniki swego działania wysyłają na strumień 2 (np. LIST itp.).

Do kontroli stanu alokacji strumieni służy rozkaz *LIST#*. Badanie napotkania końca pliku można realizować dwojako:
- poprzez mechanizm ON ERROR GOTO
- przez rozkaz IF# nr strumienia, numer linii, do której należy skoczyć, jeśli został odczytany ostatni element pliku.

ROZSZERZENIA JĘZYKA BASIC ZWIĄZANE Z DODANIEM NOWYCH ROZKAZÓW EDYCYJNYCH

Edytor języka BASIC został rozszerzony o rozkazy:
RESTORE n1, n2 – przenumerowanie linii programu od numeru n1 do n2 wraz z instrukcjami GOTO, GOSUB itp.
LINE n1, n2 – usunięcie linii programu od numeru n1 do n2.

OPROGRAMOWANIE SYSTEMU HASEŁ

Wszystkie zbiory mogą być chronione hasłem o maksymalnej długości 9 znaków. System pamięta hasła 7 użytkowników. Hasła te są ustawione w kolejności, która wyznacza stopień uprzywilejowania danego użytkownika. Właściciel dyskietki (osoba, która ją formatowała i podała swoje hasło) ma wgląd do wszystkich zbiorów na dyskietce. Osoba, która uzyska drugie w kolejności hasło będzie miała wygląd do zbiorów wszystkich osób oprócz właściciela itd.

OPROGRAMOWANIE OBSŁUGI PRZYCISKU NMI (PRZERWANIE NIEMASKOWALNE)

Jak wiadomo w pamięci ROM mikrokomputera ZX Spectrum znajduje się błąd, który uniemożliwia korzystanie z przerwań niemaskowalnych (tzn. takich, których nie można zablokować programowo) mikroprocesora Z80.

System MUEL-85 koryguje ten błąd i daje nowe możliwości, takie jak:
- kopia ekranu w dowolnym momencie i powrót do programu
- wyjście do poziomu języka BASIC bez kasowania pamięci
- wyjście do poziomu języka BASIC przez wpisanie do zmiennej systemowej RAMTOP adresu 23 999 i wykonanie instrukcji NEW
- skok wg wektora umieszczonego w nowych zmiennych systemowych, zawartych w dodatkowym 1 KB pamięci RAM.

INNE ROZSZERZENIA

System MUEL-85 zawiera następujące nowe rozkazy języka BASIC:
CODE a, b, c – przepisanie bloku pamięci rozpoczynającego się od adresu *a* i o długości *c*, pod adres *b*
GOTO★ a – skok do podanej linii w wypadku wystąpienia błędu
PEEK a, *b* – listowanie zawartości pamięci od adresu a do adresu *b*; drukowane są: zawartość komórki jako bajt (liczba 8-bitowa), jako słowo (liczba 16-bitowa), oraz znak wg kodu ASCII
CLS★ – przywrócenie początkowych kolorów tła, znaków i ramki
POKE★ a,b★ – wpisanie 16-bitowej liczby pod adresem a oraz a + *1*.

W oprogramowaniu systemowych interfejsu przewidziano możliwość przyłączenia RAM-dysku, czyli dużej ilości pamięci półprzewodnikowej symulującej piąty dysk. Wielką zaletą takiego urządzenia jest bardzo krótki czas dostępu do pamięci zewnętrznej.
Istnieje następujące oprogramowanie użytkowe współpracujące z interfejsem:
- zmodyfikowana wersja makroasemblera GENS3M21
- asembler 8049/8048/8035
- edytor tekstowy
- program kalkulacyjny dla firm rzemieślniczych.

Nazwa: MUEL-85
Typ: uniwersalny interfejs, system dyskowy
Producent:
Zakład Rzemieślniczy MUEL, Warszawa
Wymiary:
25×16,5×2,5 cm
Sterownik dysku Western Digital FD1791
Pamięć: 10 KB ROM, 1 KB RAM
Możliwość dołączenia: 4 napędów dysków o liczbie ścieżek od 3 do 255
1 drukarki dowolnego typu
1 monitora monochromatycznego
Zasilacz: sieciowy 220V/50Hz na 12V, 5V, 9V
Oprogramowanie: dyskowy system operacyjny i rozszerzony BASIC w pamięci ROM

PIOTR PARLEWICZ
ANDRZEJ WIĘCKOWSKI
GRZEGORZ ZAWADZKI

Pani dr Barbara Chrzan-Feluch, autorka poniższego artykułu, jest pracownikiem naukowym Instytutu Matematyki Uniwersytetu Warszawskiego. Odbyła staż naukowy w uniwersytetach Paryża i Bordeaux, uczestnicząc w seminariach poświęconych nauczaniu informatyki w przedszkolach i szkołach francuskich.
Wydaje się, że Jej spostrzeżenia mogą być interesujące nie tylko dla nauczycieli, ale także dla producentów zabawek i pomocy szkolnych.

Komputer czy cyrkiel i linijka

W 1980 r. na prośbę prezydenta Giscard d'Estaing powstał raport przygotowany przez J.C. Simon nt. „Edukacja i informatyzacja społeczeństwa". Czytamy w nim m.in.: *„Edukacja informatyczna jest konieczna, ponieważ informatyka jest ważnym zjawiskiem w procesie poznawczym, a jej idee przenikają do innych dyscyplin, które potrzebują jej metod i języka, aby mogły być konsekwentnie wykorzystywane przez nasze społeczeństwo. Trzeba więc uczyć jej wszystkich młodych Francuzów."*

W ubiegłym roku szkolnym obserwowałam we Francji wyniki tej akcji. Zaczyna się często już od przedszkola. Informatyka wchodzi tu głównie za pośrednictwem zabawek-robotów, takich jak: „Big-Trak", „Żółw", pojazdy cybernetyczne czy ciężarówki programowane.

W jednym z paryskich przedszkoli oglądałam, razem z grupą przybyłych z Kanady nauczycieli, pokazowe zajęcia. Prowadziła je Catherine Berdonneau pracująca w „Grupie Badań i Studiów nad Praktyką i Koordynacją Informatyki w Kształceniu Początkowym i Ustawicznym". Dzieci, razem ze swoją wychowawczynią, siedziały wokół niedużego aparatu połączonego z głośnikiem. W niewielkie pudełko wkłada się plastykowe karty – rozkazy dla robota. Program układa się z takich rozkazów, jak: „Graj do", „Graj re", „Graj szybciej", „Powtórz jeszcze raz", itp. Dzieci są bardzo zainteresowane. Podoba im się, że każdy rozkaz powoduje głośną reakcję robota. Zadanie polega na ułożeniu programu znanej piosenki „Frère Jacques".

Na innych zajęciach dzieci „uczą" pojazd kosmiczny o kształcie szklanej półkuli rysować koło. Pojazd ten przesuwając się zaznacza przebytą drogę mazakiem na papierze. Dzieci wkładają kolejne karty-rozkazy i sprawdzają, co robi pojazd. Jeden z rozkazów powoduje obrót w prawo. Sprawdzamy narysowany ślad drogi – jest to 1/4 koła. Ile razy trzeba powtórzyć ten rozkaz, żeby narysować całe koło? Niektóre dzieci mówią dwa, inne trzy, inne cztery razy. Sprawdzamy eksperymentalnie, kto miał rację. W ten sposób dzieci uczą się pojęcia koła trochę inaczej niż w tradycyjnej dydaktyce.

Kolejny przykład dotyczy dzieci opóźnionych w rozwoju. Zajęcia są prowadzone w klasie specjalnej. Uczniowie, 7-10-letni, mają kłopoty z mówieniem i liczeniem. Na lekcję nauczycielka przynosi elektronicznego żółwia. Naciśnięcie odpowiednich guziczków, znajdujących się na jego grzbiecie, powoduje wykonanie rozkazów: *CM* – słuchaj mnie, *↑2* – idź dwa kroki do przodu, *↓3* – idź trzy kroki do tyłu, *GO* – ruszaj, itp. nauczycielka rysuje kredą na podłodze drogę, oddzielając kolejne kroki żółwia. Następnie zadaje pytanie:
O ile kroków przesunie się żółw do przodu, gdy wpiszemy program:

CM ↑3 ↓1 GO

O dwa kroki, bo trzy kroki do przodu i jeden do tyłu to tak, jakby dwa do przodu (3 – 1 = 2).
Teraz nauczycielka poleca ułożyć krótszy program, a uczeń pisze na tablicy:

VM ↑2 GO

W ten sposób nauczycielka uczy dzieci liczenia, sensu działań arytmetycznych. Był to oczywiście tylko jeden ze stosowanych sposobów. Chodziło o zainteresowanie dzieci problemem, o skupienie ich uwagi. Przykład działania *3–1=2* jest dla dzieci może mało ciekawy, ale ciekawe dla niego jest to, co zrobi żółw.

W eksperymentalnej szkole Uniwersytetu w Bordeaux znany francuski dydaktyk, Guy Brousseau, sprawdza opracowane ze swoim zespołem programy mikrokomputerowe. Uważa, że programy, które można kupić w sklepach, są pod względem dydaktycznym bardzo słabe.

Obserwujemy tu zajęcia z dziećmi 4-5-letnimi. Na ekranie kolorowego telewizora, do którego przyłączony jest mikrokomputer TO-7, pojawiają się króliki. Należy je „przewieźć" w łódkach na drugą stronę rzeki. Dla każdego królika potrzebna jest jedna łódka. Aby ona powstała wystarczy dotknąć piórem świetlnym ekranu w miejscu, gdzie jest rzeka. Trzeba przygotować dokładnie tyle łódek, ile jest królików. Jak to zrobić, gdy się ma 4-5 lat i jeszcze nie umie dobrze liczyć. Dzieci jednak jakoś sobie radzą. Niektóre zapisują na kartce: 1,2,3,5,4,6 ... i później liczą łódki w tej samej kolejności. Inne zaznaczają króliki krzyżykami i następnie skreślają po jednym krzyżyku. Bardzo cieszą się, gdy uda im się przewieźć wszystkie króliki.

Dla uczniów jeszcze starszych (11 lat) przygotowuję razem z Brousseau program „Ogrodnik", dotyczący proporcjonalności. Na wspaniałe, wyhodowane przez ogrodnika kwiaty „napadają" owady. Trzeba więc zastosować truciznę, którą stosuje się w różnych proporcjach, np. jedna kulka na dwa owady, następnie trzy kulki na pięć owadów itp. W tym celu należy zamówić odpowiednią liczbę kulek. Każda pomyłka kończy się fatalnie dla kwiatów. Gdy trucizna nie zniszczy wszystkich owadów zjadają one kwiaty. Jeżeli trucizny jest za dużo, to kulki trucizny niszczą kwiaty. Komputer, wykonując odpowiedni rysunek, wyświetla informację: „Stosować w proporcjach 3 kulki na 5 owadów". Uczeń jest mobilizowany do uważnego odczytania tej informacji: szkoda mu bowiem kwiatów, a jednocześnie chciałby wygrać. Czynniki te powodują szybsze i dokładniejsze, bo bardziej aktywne uczenie się.

Ciekawy sposób włączenia mikrokomputera w tradycyjne nauczanie matematyki proponuje Serge Hocquenghem z Uniwersytetu Paris 7. Mikrokomputer podawał jakiś interesujący fakt matematyczny, np. składał symetrie osiowe, a uczniowie mieli zadanie fakt taki zauważyć i udowodnić. Oto przykład tego rodzaju programu, opartego na pomyśle Hocquenghema, który opracowałam na mikrokomputer ZX Spectrum.

Komputer wykonuje odbicie symetryczne względem pierwszej osi, następnie drugiej, potem znów pierwszej, itd. (patrz rysunek). Punkty układają się na okręgu (dlaczego?). Składając symetrie osiowe o osiach przecinających się otrzymujemy obrót.

Wielką pomocą w rozwijaniu informatyki w szkołach francuskich jest Centrum Informatyki Ministerstwa Oświaty. Znajdują się w nim różne typy obecnie używanych mikrokomputerów szkolnych wraz z biblioteką i katalogami programów edukacyjnych, które można tu obejrzeć i wypróbować. Wspomniane Centrum prowadzi również zajęcia dla dzieci z zastosowaniem mikrokomputerów.

LOGO

Martha Mahieu w pracy pt. „Dzieci programowane czy dzieci programujące" pisze o swoich doświadczeniach dotyczących wykorzystania w nauczaniu w jednej z belgijskich szkół języka Logo. Rozróżnia w niej nauczanie wspomagane przez komputer oraz uczenie się wspomagane przez komputer. W pierwszym wypadku maszyna programuje ucznia, w drugim – uczeń programuje maszynę.

Przypomnijmy, że Logo jest językiem programowania opracowanym dla dzieci. Jednym z jego twórców jest J. Papert, który przez pewien okres współpracował z J. Piagetem w znanym ośrodku psychologii w Genewie. Piaget, jeden z największych psychologów badających rozwój umysłowy dzieci, wyróżnił w tym procesie kilka etapów. Dokonując setek obserwacji zachowania dzieci, rozwiązujących przygotowane dla nich zadania, potrafił dokładnie scharakteryzować sposób myślenia dzieci, typowy dla poszczególnych etapów ich rozwoju. Wielu pedagogów podejmowało próby przyspieszenia tego rozwoju, ale bez istotnych rezultatów. Wydaje się, że nie udało się tego dokonać również drogą Logo. Nie potwierdzono też eksperymentalnie hipotezy Paperta, że *„... Logo przez manipulację mikroświatem jednocześnie konkretnym i systematycznym, może przyspieszyć u dzieci rozwój operacji formalnych"*.

Z punktu widzenia celów nauczania ważne jest nie tyle przyspieszenie w rozwoju dziecka momentu, w którym będzie ono umiało myśleć w sposób formalny, co wszechstronny jego rozwój, umożliwiający w przyszłości korzystanie oraz dalsze rozwijanie istniejącego stanu wiedzy. Obecni uczniowie spotkają się w przyszłości z powszechnym stosowaniem komputerów i dlatego wczesne oswajanie ich z myśleniem informatycznym jest dziś bardzo potrzebne.

Eksperyment Marthy Mahieu był prowadzony z dziećmi 9-10-letnimi. Zajęcia trwały pół godziny: prowadzone były na zmianę z całą grupą, a następnie w zespołach 2-3-osobowych z mikrokomputerem. Dotyczyły głównie rysowania coraz bardziej złożonych figur geometrycznych. Poszczególne instrukcje języka Logo były zapisane i przedstawiane dzieciom na kartkach. Cenna jest obserwacja Mahieu: *„dzieciom należy przekazywać jak najmniejsze porcje informacji i następnie stosować ćwiczenia, w których informacje te mogą praktycznie wykorzystać"*. Często bowiem wpada się w pułapkę, polegającą na pokazaniu wszystkiego, co można zrobić (zmienne, operacje logiczne itp.).

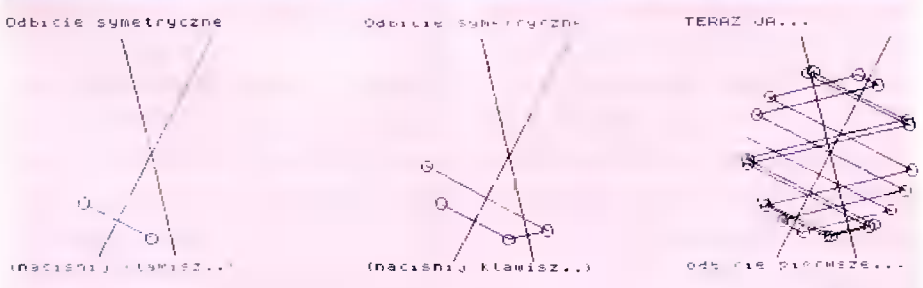
Mahieu podaje następujący przykład. Dzieci rysowały koła o różnych promieniach. Każde koło było opisane w oddzielnych instrukcjach. Mahieu zaproponowała asystentowi wprowadzenie zmiennej, uznając, że jest to właściwy moment. Ale asystent odpowiedział: *„O nic nie pytają, zaczekaj"*. Dziesięć minut później dzieci powiedziały: *„To głupio pisać dziesięć razy taki sam program, tylko z inną liczbą. Czy nie ma jakiejś sztuczki, której możemy tu użyć?"*.

Logo można wykorzystać również w nauczaniu starszych dzieci. Realizowano np. takie tematy, jak wykresy funkcji, badanie podzielności liczb, rozwiązywanie równań I i II stopnia, przekształcenia geometryczne, miara (przybliżanie miary koła jako granicy miar wielokątów), opracowywanie banków informacji.

Na jednym z seminariów uniwersyteckich, w Paryżu słuchałam referatu dwóch nauczycielek z Kanady, które prowadziły eksperyment porównawczy w dwóch klasach liceum. W jednej klasie pracownia była wyposażona w mikrokomputery Commodore i program realizowano opierając się na Logo (m.in. funkcje, rachunek różniczkowy i całkowy, miarę). Te same tematy realizowano w klasie równoległej, ale bez pracowni mikrokom-

Wydruk programu w języku BASIC

```
  0 REM "ODBICIE "
 10 LET i=0: LET j=0
 20 LET x=INT (RND*100)+20: LET y=INT (RND*100)+20
 30 IF (x-100)*(x-100)+(y-80)*(y-80)>3600 THEN GO TO 20
 40 CIRCLE x,y,4
 50 LET a=INT (RND*200): LET b=INT (RND*160)
 60 PLOT a,160: DRAW 200-2*a,-160
 70 PLOT b,160: DRAW 200-2*b,-160
 80 LET c=((100-a)*(y*80-160*80)-x*(100-a)*(100-a)-a*6400)/(-64
00-(100-a)*(100-a))
 90 LET d=(((c-x)*(100-a))/80)+y
100 LET xi=c+((80*(d-y))/(100-a))
110 LET yi=2*d-y
120 LET j=j+1
130 IF i=1 THEN GO TO 240
140 IF j>5 THEN GO TO 170
150 PRINT AT 0,0;"Odbicie symetryczne:";AT 21,0;"(nacisnij klaw
isz..)": PAUSE 0
160 GO TO 180
170 PRINT AT 0,0;"TERAZ JA...              ";AT 21,0;"odbicie pie
rwsze...      ": FOR t=1 TO 10: BEEP .02,t: NEXT t
180 PLOT x,y: DRAW x1-x,yi-y
190 BEEP 0.1,0.2: CIRCLE x1,yi,4
200 LET x=xi: LET y=yi: LET i=1
210 LET z=a: LET a=b
220 GO TO 80
230 PRINT AT 21,0;"TERAZ JA..."
240 IF j>5 THEN GO TO 270
250 PAUSE 0
260 GO TO 280
270 PRINT AT 21,0;"odbicie drugie..."
280 BEEP 0.1,0.7: CIRCLE x1,y1,4
290 IF j>60 THEN GO TO 340
300 PLOT x,y: DRAW x1-x,y1-y
310 LET x=x1: LET y=y1: LET i=0
320 LET a=z
330 GO TO 80
340 PRINT AT 0,0;"Co to jest ?";AT 21,0;"...dlaczego tak jest?"
: STOP
```



Rysunek otrzymany na ekranie monitora

puterowej. Po rocznym eksperymencie przeprowadzono typowy test, który zwykle przeprowadza się tam pod koniec roku szkolnego. Statystyczne porównanie wyników testu obydwu klas nie wykazało między nimi istotnych różnic. Niektórzy uczestnicy seminarium stwierdzili, że wynik jest korzystny dla Logo, ponieważ uczniowie poza opanowaniem podstawowego programu nauczyli się trochę informatyki, czego test już nie sprawdzał. Inni uważali, że nie ma sensu uczyć języka, w którym żaden dorosły nie programuje. Basic nie jest dużo trudniejszy, a w nowych rozwiązaniach mikrokomputerów rysowanie jest coraz łatwiejsze. Zresztą nie jest ważny język, ważne są metody informatyki.

MIKROKOMPUTERY W NASZYCH SZKOŁACH

Chociaż niektórzy mówią, że w naszych szkołach nie ma dobrej kredy ani podręczników, więc co tu mówić o mikrokomputerach, w rzeczywistości jednak coraz liczniejsze są szkoły wyposażone w mikrokomputery. Jestem pewna, że wkrótce będzie ich w szkołach tyle, ile jest już tam telewizorów.

Wiele zagranicznych uczelni technicznych niegdyś stawiało studentom warunek posiadania i umiejętności liczenia na kalkulatorze. Dzisiaj wymagają one posiadania mikrokomputera, co umożliwia realizację zupełnie innego programu nauczania.

Wydaje się, że również u nas najwyższy już czas przygotować dla szkół mikrokomputerowe programy edukacyjne oraz przystępną literaturę zarówno dla uczniów, jak i nauczycieli. Wyposażenie szkół w mikrokomputery powinno iść w parze z przygotowaniem odpowiedniego oprogramowania, warunkującego ich efektywne zastosowanie. Czy CEZAS zajmie się sprzedażą mikrokomputerowych programów edukacyjnych? Czy powstanie u nas, podobnie jak w Paryżu, Centrum Informatyki dla Szkół, które gromadziłoby programy i literaturę? Nauczyciele i uczniowie mają często doskonałe pomysły programów edukacyjnych, ale nie mają czasu, aby je opracować w formie nadającej się do rozpowszechnienia. Czy nie warto dopracować tych pomysłów oraz zachęcić autorów do ich opisania i przesłania np. do proponowanego Centrum? Na razie więcej tu pytań niż odpowiedzi.

Najbliższa konferencja Szkoły Dydaktyki Matematyki, która odbędzie się latem w Sierpii, będzie poświęcona mikrokomputerom w szkołach. Zapraszamy z pomysłami i komputerami.

BARBARA CHRZAN-FELUCH

Mikrokomputer zamiast deski kreślarskiej

Projektowanie wspomagane mikrokomputerem jeszcze przed kilku laty było nie do pomyślenia. Dzisiaj jest już inaczej. Pojawiło się wiele mikrokomputerowych systemów CAD (ang. Computer Aided Design – projektowanie wspomagane komputerem) o przystępnej cenie. Wybor nie jest prosty. Spróbujmy więc określić, jakie cechy musi mieć dobry system i na co należy zwracać uwagę przy zakupie.

Dotychczas stosowano dwa sposoby organizacji projektowania wspomaganego komputerem. Z jednej strony używane były specjalizowane minikomputery z kilkoma stanowiskami pracy, z drugiej zaś – duże uniwersalne komputery zainstalowane w ośrodku obliczeniowym. Projektanci uważają to drugie rozwiązanie za mniej wygodne, gdyż proces konstruowania musi być wtedy zharmonizowany z pracą całego ośrodka obliczeniowego, co powoduje istotne ograniczenie dostępu do systemu. Z kolei korzystanie z systemu CAD eksploatowanego na specjalizowanym minikomputerze prowadzi często do zbyt wysokich kosztów sprzętowych i programowych. Kompromisem między wydajnością a kosztami są systemy CAD oparte na mikrokomputerach. Pod względem wielkości nakładów inwestycyjnych są one bez wątpienia konkurencyjne w stosunku do minikomputerów. Znajdują zastosowanie w tych dziedzinach, gdzie ich wydajność jest wystarczająca.

SPRZĘT

Wiele małych firm oferuje dziś gotowe systemy CAD. Ich zaletą jest to, że zarówno sprzęt, jak i oprogramowanie pochodzą z tej samej firmy oraz że system jest produktywny natychmiast po zainstalowaniu. W wyrobach tych na ogół stosuje się mikroprocesor 8088 lub 8088, wspomagany przez współprocesor arytmetyczny 8087.

Najważniejszym składnikiem sprzętu jest monitor graficzny, w którym wymagany jest kolor i wysoka rozdzielczość. Tymczasem rozdzielczość standardowych monitorów komputerów osobistych pozostawia często wiele do życzenia. Istnieją więc systemy pracujące z dwoma monitorami, alfanumerycznym i graficznym. Specjalne moduły sprzętowe umożliwiają adaptację monitorów graficznych. Monitor dobrej jakości może jednak decydująco wpłynąć na koszt całego systemu.

Do wprowadzania poleceń służy pulpit graficzny. W mikrokomputerowej instalacji CAD jest on niezbędny. W urządzeniu tym operator przesuwa manipulator z celownikiem w płaszczyźnie *xy* po powierzchni pulpitu i przez wciśnięcie odpowiedniego przycisku powoduje automatyczne wprowadzenie współrzędnych z celownika do komputera. Urządzenia o podobnym przeznaczeniu – pióro świetlne i myszka – są mniej wygodne i ich stosowanie nie jest zalecane.

Trwałe, dobrej jakości kopie obrazów z ekranu monitora można uzyskiwać na ploterach o formacie A4 i A3, charakteryzujących się niezbyt wysoką ceną. Plotery o formacie A0, zapewniające dużą rozdzielczość i odpowiednio wysoką jakość rysunku, są i pozostaną w przyszłości drogie, z powodu swej skomplikowanej budowy mechanicznej.

EDYTOR RYSUNKÓW

Systemy CAD stosowane są w coraz szerszym zakresie. Ułatwiają, a często w ogóle umożliwiają, szybkie przygotowanie dokumentacji ofertowej, opracowanie koncepcji i wykonanie projektów, kreślenie rysunków, sporządzanie zestawień elementów, opracowanie planów pracy, programowanie obrabiarek sterowanych numerycznie i sprawdzanie jakości wyrobów.

Podstawowym środkiem porozumiewania się w całym procesie produkcyjnym jest rysunek techniczny. Stąd szczególne znaczenie ma wspomagane komputerem wykonywanie rysunków. Proces ten powinien zapewniać powiązanie z programami wykorzystania zawartości rysunków w dalszych etapach produkcji.

W systemie dwuwymiarowym podstawowymi elementami geometrycznymi są: punkt, prosta i okrąg. Za pomocą odpowiednich operacji można z tych elementów tworzyć opisy geometryczne obiektów technicznych. Możliwe są różnorodne odmiany poleceń. Na przykład proste polecenie: *utwórz odcinek*, po modyfikacji może mieć następującą złożoną postać: *utwórz odcinek równoległy do danego innego odcinka i przechodzący przez dany punkt*.

Wydajność oprogramowania zależy w istotnym stopniu od wielkości zbioru poleceń, możliwości ich użycia i kombinowania w różnych zastosowaniach. Istotny jest też sposób i systematyka stosowanego opisu obiektów technicznych. Z podstawowych elementów geometrycznych tworzone są figury definiowane przez użytkownika. Mogą one być zapamiętane w bibliotekach graficznych (w pamięci zewnętrznej) i później wielokrotnie używane, jako indywidualne elementy użytkowe.

Geometria części zaprojektowanej w czasie dialogu graficznego zostaje przetransformowana do postaci, w jakiej informacje przechowywane są wewnątrz komputera. Wewnętrzna struktura danych musi spełniać wysokie wymagania. Czasy dostępu do informacji, nawet przy skomplikowanych figurach, nie mogą być zbyt długie. Niezbędna jest możliwość wprowadzania zmian. Wprowadzanie zmian w rysunku musi pociągać za sobą odpowiądające im zmiany w danych, opisujących ten rysunek. Jest to istotne – na przykład – przy zakreskowaniu konturów lub wymazywaniu części rysunku.

STRUKTURY DANYCH

Tak więc struktura danych ma bezpośredni wpływ na wydajność oprogramowania systemu **CAD.** Jako przykład takiego oprogramowania opiszemy pakiet o nazwie *Memoplot*, zainstalowany na mikrokomputerze Sirius, wyposażonym w monitor o rozdzielczości 800×400 punktów. Odpowiada to jakości obrazu specjalnych monitorów graficznych i dlatego umożliwia szerokie stosowanie grafiki komputerowej.

Po zakończeniu etapu tworzenia rysunku zostaje on zapamiętany w komputerze w postaci cyfrowej. Można go wyprowadzić w postaci tradycyjnej na papierze, za pomocą graficznego urządzenia końcowego, najczęściej plotera.

Każdy rysunek składa się z wielu części. Częścią rysunku może być np. ramka, figura złożona z innych figur lub powiększony wycinek jednej z figur. Część rysunku jest najwyżej zorganizowaną jednostką struktury danych. W czasie pracy z edytorem graficznym, jest ona identyfikowana przez nazwę zapisaną w skorowidzu zbiorów. W procesie projektowania rysunek jest składany z oddzielnych części.

Dla uzyskania przejrzystości wprowadzono strukturalizację danych poprzez utworzenie warstw lub płaszczyzn. Poszczególne linie jednej figury mogą należeć do różnych warstw. Rysunek można tworzyć z dowolnych warstw. Rodzaj linii, grubość kreski, kolor itp. mogą być definiowane warstwami. Po opracowaniu części rysunku następuje identyfikacja elementów graficznych, jedynie w warstwach określonych jako aktywne.

Figura jest strukturą ponadwarstwową. Można ją definiować jako składającą się z dowolnych elementów bazy danych. Uaktywnienie określonych warstw powoduje wyświetlenie tych części figury, które należą do tych warstw, co daje szerokie możliwości manipulowania. Przy takim podejściu, nawet przy bardzo wielkiej liczbie danych, czas odpowiedzi systemu jest możliwy do zaakceptowania.

WPROWADZANIE ROZKAZÓW

Wprowadzanie rozkazów jest możliwe zarówno z klawiatury, jak i za pomocą pulpitu graficznego. Wszystkie polecenia są uporządkowane w postaci hierarchicznej struktury drzewiastej. Dzięki temu, na każdym etapie pracy użytkownik ma do dyspozycji niewielki i przez to przejrzysty zestaw poleceń. Szczególne znaczenie ma menu dotyczące wprowadzania punktu, które jest dostępne na każdym etapie. Różne funkcje wymagające wprowadzenia współrządnych jednego lub kilku punktów można wywołać przez odpowiednie ustawienie manipulatora z celownikiem i wciśnięcie jednego z kilku umieszczonych na manipulatorze klawiszy. Jeśli, na przykład, chcemy wprowadzić punkt pośredni leżący na odcinku, to wystarczy naprowadzić kursor w jego pobliże i wcisnąć klawisz S celownika. W ten sposób zostaną wprowadzone współrzędne punktu leżącego na odcinku i znajdującego się najbliżej położenia celownika. Jedna z dwóch części, na które został podzielony odcinek, może teraz być poddana dowolnym operacjom, na przykład można ją wymazać.

Do funkcji manipulacyjnych należą transformacje – zwykłe i wielokrotne. Figury można dowolnie transformować przez podanie rodzaju transformacji oraz wybranie punktów odniesienia na samych figurach i odpowiadających im punktów w tym miejscu rysunku, do którego figura ma być transformowana. Transformacjami mogą być m.in. przesunięcie, obrót i odbicie lustrzane. Jeśli indywidualnie skonstruowana figura ma być narysowana wielokrotnie, to używa się w tym celu transformacji wielokrotnej, którą można oddzielnie realizować w postaci wspomnianych przesunięć, obrotów i odbić. Po wyzwoleniu transformacji wielokrotnej mogą być przeprowadzone wydzielone manipulacje na częściach figur.

Dla scharakteryzowania kształtu i podania innych informacji o obiekcie w rysunku technicznym, istotne są wymiarowanie i opis. System CAD powinien umożliwiać jak najdalej zautomatyzowane wymiarowanie zgodne z obowiązującymi normami. Istotna jest możliwość umieszczenia w dowolnym miejscu rysunku napisów wykonanych znormalizowanym pismem technicznym różnego typu o wybranej wysokości, pochyleniu czy grubości. Dla ułatwienia procesu projektowania stosuje się siatkę kwadratową (raster) o dowolnie wybranych odstępach w kierunkach *x* i *y*. Możliwe jest również stosowanie rastrów prostokątnych.

Na wszystkich etapach projektowania bardzo przydatne są takie funkcje systemowe jak: zmiana skali przez powiększenie lub pomniejszenie obrazu (ang. zoom), tworzenie okienek w układzie współrzędnych, możliwość uzyskania w każdej chwili trwałej kopii. Dużą pomocą są funkcje informacyjne, które śledzą stan systemu. Polecenie *HELP* (ang. pomocy!) umożliwia uzyskanie w czasie wykonywania programu istotnych informacji operatorskich.

Podsumowując należy stwierdzić, że wejście w zastosowania CAD nie stanowi dziś dla potencjalnego użytkownika komputera osobistego praktycznie żadnego ryzyka. Dalszy rozwój techniki spowoduje wkrótce wyeliminowanie w wyniku konkurencji tych osób, które nie będą korzystały w swej pracy zawodowej z CAD.

micro

dokończenie ze str. 21

AMSTRAD CPC 6128

ciej niż magnetofon. Sama dyskietka kosztuje ok. 7 dolarów i ma pojemność 2×180 KB, co powoduje, że zapamiętywanie na dyskietkach 5 1/4-calowych jest trzy razy tańsze. Należy jednak spodziewać się, że minidyskietki będą stopniowo taniały.

SYSTEMY OPERACYJNE

Instrukcje systemu operacyjnego AMS DOS są dostępne z poziomu języka BASIC. Możliwe jest wprowadzanie, zapamiętywanie, kasowanie i przemianowywanie plików (zbiorów danych). System ten wspomaga również obsługę magnetofonu kasetowego.

Przestawienie na system CP/M, odbywa się za pomocą instrukcji CPM. System zarówno w wersji 2.2, jak i nowszej 3.1 (lub CP/M+) jest dostarczany wraz z komputerem. Wersja 2.2 może być interesująca, jeśli zamierza się użytkować starsze oprogramowanie. Wygodniejsza w zastosowaniu jest wersja 3.1, pod nadzorem której CPC 6128 udostępnia obszar pamięci operacyjnej TPA (Transient Program Area) o pojemności 61 KB. Stwarza to możliwość wykorzystania całego oprogramowania CP/M-80 (WordStar, Supercalc2, Multiplan itp.). Wersja 3.1 daje również możliwość wyboru ośmiu narodowych zestawów znaków pisarskich.

Objęte dostawą programy: uruchomieniowy (ang. debugger), asembler oraz testujący (DDT) wspomagają niestety mikroprocesor 8080 i w zastosowaniu do Z80 nie spełniają swych funkcji. Jest więc niezbędna aktualizacja tych programów. Edytor ED pochodzi jeszcze z okresu pionierskiego mikrokomputerów i również wymaga modernizacji. Możliwość przetwarzania wsadowego (ang. batch) pod nadzorem CP/M przybliża z kolei CPC 6128 do kategorii komputerów osobistych.

BASIC I LOGO

Aby zapewnić kompatybilność z poprzednimi modelami pozostawiono niezmieniony BASIC – stąd jego rozszerzenia dla CPC 6128 wprowadzane są z dyskietki. Jak wiadomo 16-bitowa szyna adresowa Z80 umożliwia bezpośrednie adresowanie tylko 64 KB. Do zarządzania przestrzenią 128 KB dostarczany jest więc na dyskietce dodatkowy program, który zapewnia korzystanie z pozostałych 64 K. Dla tekstu programu w języku BASIC dysponować można, podobnie jak w modelach poprzednich, obszarem ok. 41 KB.

Do wyposażenia komputera należy również wersja języka Logo o nazwie *Dr. Logo*. Jest to implementacja specjalna wspomagająca syntezator oraz klawisze kursora. Wersja ta została znacznie rozszerzona i obecnie całkowicie odpowiada Logo firmy Digital Research. Dla CPC 6128 przygotowano również własną wersję Turbo-Pascala. Zwiększony obszar pamięci zapewnia, że nie są już potrzebne (jak w innych modelach) żmudne czynności dzielenia programów na części.

Jeśli użytkownika interesuje muzyka, to istnieje możliwość komponowania na trzy głosy, w ośmiu oktawach. Programista może ustalać własną intonację i modulację. System syntezatora umożliwia ustawienie w kolejce do pięciu instrukcji sterujących dźwiękiem w każdym kanale.

Do wymienionego oprogramowania dołączono również program grafiki GSX. Umożliwia on wyprowadzenie tekstu połączonego z grafiką na ekran, drukarkę lub ploter. Jednak GSX można zastosować tylko wtedy, gdy zostaną użyte programy wspomagające (np. Dr. Graph, firmy Digital Research).

RYNEK OPROGRAMOWANIA

Producent CPC 6128 oferuje stosunkowo skromne oprogramowanie użytkowe (przetwarzanie tekstów, tabelaryczne zestawienie kalkulacyjne, zarządzanie zapasami, języki, gry). Wykorzystują to liczni producenci oprogramowania, którzy usiłują partycypować w sukcesach rynkowych Amstrada, oferując np. okrojone wersje takich szlagierów, jak WordStar, dBase i Multiplan.

Jak wiadomo, dla systemu operacyjnego CP/M oprogramowania nie brakuje. Koncentruje się ono jednak zwyczajowo na dyskietkach 8- lub 5 1/4-calowych. Jego przeniesienie na Amstrada poprzez dodatkowy napęd dyskietek jest w zasadzie możliwe, ale tylko dla doświadczonych użytkowników. Biorąc jednak pod uwagę obecną mocną pozycję Amstrada na rynku mikrokomputerowym należy liczyć się z tym, że wkrótce masa oprogramowania CP/M będzie dostępna również na dyskietkach 3-calowych.

Za umiarkowaną cenę Amstrad oferuje łącznie z komputerem dobrą dokumentację. Dzięki znacznym rozmiarom produkcji modelu 6128 spodziewać się należy intensywnego rozwoju oprogramowania. Jak już wspomniano, system operacyjny CP/M otwiera dostęp do wielkiej światowej biblioteki oprogramowania.

Mimo wielu zalet CPC 6128 znajduje odbiorców nadal głównie w środowisku hobbistów i młodzieży, ponieważ jego ekran nie spełnia wymagań zastosowań profesjonalnych. Dodatkowymi przeszkodami dla tych zastosowań jest brak pełnowartościowego interfejsu drukarki oraz klawiatury zgodnej z normą maszyn do pisania, nie wspominając już o braku bardziej pojemnej pamięci dyskowej.

Gwałtowny spadek cen komputerów kompatybilnych z IBM PC może stworzyć wkrótce dla Amstrada znaczne trudności rynkowe. Wyposażony tylko w dwie stacje dyskowe, CPC 6128 prędzej czy później zderzy się z dolną granicą cen sprzętu kompatybilnego. Jedynie wysokiej klasy nowatorskie oprogramowanie może dopomóc Amstradowi odeprzeć atak tajwańskiej konkurencji.

micro PIOTR SMÓLSKI

dokończenie ze str. 20

Pamięć dynamiczna dla procesora Z80

generując jednocześnie sygnały strobujące *MREQ* i *RFSH*. Dzięki temu proces odświeżania pamięci dynamicznej staje się zupełnie niewidoczny dla procesora i nie wymaga spowalniania jego pracy przez generowanie sygnału *WAIT*.

Rysunek 1 przedstawia realizację układu sterowania pamięcią dynamiczną przy użyciu układów o średnim stopniu scalenia. Sterownik pamięci składa się z trzech zasadniczych elementów:

- multipleksera adresów
- generatora sygnałów strobujących *RAS, MUX, CAS*
- dekodera przestrzeni adresowej.

Multiplekser adresów służy do podawania na matrycę pamięci adresu w dwóch fazach, najpierw adresu kolumn, później adresu wierszy.

Sygnał *RAS* generowany jest do pamięci podczas:

- dostępu do pamięci przez procesor
- cyklu odświeżania.

W pierwszym wypadku sygnał strobujący *RAS* wystawiany jest wówczas, gdy zostaną spełnione dwa warunki: pamięć zostanie wybrana (odpowiednie wyjście dekodera adresów 74LS138 będzie w stanie aktywnym) oraz procesor wystawi sygnał strobujący *MREQ*. Ponieważ adres na szynie adresowej ulega zmianie w czasie trwania sygnału *MREQ* (rys. 2), istnieje możliwość pojawienia się krótkotrwałego impulsu na wejście *RAS* matrycy pamięci. W celu wyeliminowania tego zjawiska można zastosować na wejściach *A0, A1, A2, E0* układu 74LS138 4-bitowy port 74LS75, w którym zapamiętany jest adres na czas trwania sygnału *MREQ*.

Podczas odświeżania sygnał *RAS* jest generowany wtedy, gdy występują jednocześnie sygnały *MREQ* i *RFSH*. Sygnały *MUX* i *CAS* inicjowane są narastającym zboczem sygnału zegarowego $\emptyset_2$ i pojawiają się jedynie podczas cyklu dostępu do pamięci przez procesor (rys. 3).

W cyklu odświeżania niski poziom linii *RFSH* powoduje zablokowanie przerzutnika 74S74, uniemożliwiając pojawienie się sygnału *CAS* na wejściach matrycy pamięci.

Jak pisać do MIKROKLANU

MIKROKLAN jest dla Ciebie. Oznacza to, że i od Ciebie zależy, co będzie w nim drukowane. Będziemy wdzięczni, jeżeli napiszesz dlaczego Ci się to czy tamto nie podoba, a szczególnie wdzięczni, gdy zaproponujesz co robić, żeby było lepiej. Pamiętaj jednak, że papier nie jest z gumy i mieści się na nim skończona ilość informacji. Weź też pod uwagę prawdopodobną opinię pozostałych 99 999 Czytelników (niektórzy spośród nich są bez mała ekspertami, inni zaś stawiają pierwsze kroki...). Nadsyłane nam propozycje tematów, które zdaniem Czytelników powinny znaleźć się na łamach MIKROKLANU, kwalifikować będziemy według liczby zainteresowanych. Może właśnie Twój list przeważy szalę? Będziemy starali się reagować jak najszybciej, jednak droga od pomysłu do kiosku RUCH-u zajmuje kilka miesięcy.

Jeżeli uważasz, że wiesz coś ciekawego i chciałbyć podzielić się tym z pozostałymi Czytelnikami – napisz artykuł. Jak wskazuje nazwa naszych zeszytów pragniemy tworzyć wspólnie z Czytelnikami klan ludzi, którym zależy, aby mikrokomputery były wykorzystywane z pełną świadomością ich możliwości i ograniczeń. Możesz więc napisać o tym, co sam odkryłeś lub udoskonaliłeś, posługując się Spectrum, Commodore, Amstradem czy IBM PC (jeśli zainspirował Cię pomysł już gdzieś publikowany, powołaj się na niego, załączając do tekstu wykaz literatury). Możesz napisać o możliwościach rozbudowy lub bardziej efektywnym sposobie programowania. Możemy też opublikować Twój program, jeśli będzie on interesujący dla innych i... niezbyt długi. To samo dotyczy konstrukcji sprzętowych. Ciekawe rozwiązania układowe, niestandardowe aplikacje, czyli MIKROKLAN – forum do wymiany pomysłów i doświadczeń dla wszystkich, hobbistów i profesjonalistów, czeka na Twój głos.

Wymagania dotyczące przysyłanych materiałów:

tekst: w maszynopisie, z dwoma kopiami, (30 wierszy na stronie, 60 uderzeń w wierszu, czyli duży odstęp między liniami);

objętość: maksymalnie 7 – 9 str. maszynopisu (jasno i zwięźle); rysunki: mogą być w ołówku (przejrzyste i czytelne);

wydruki: maksymalnie kontrastowe (załóż nową taśmę do drukarki, jeśli możesz przyślij program na nośniku magnetycznym – zwrócimy!);

programy: nie tylko przetestowane, ale też eleganckie;

konstrukcje: wyłącznie uruchomione i koniecznie sprawdzone;

ponadto: adres domowy, telefony – domowy i służbowy, ew. numer konta PKO.

W następnym zeszycie

komputery IBM PC i kompatybilne